POLITECNICO DI TORINO

MASTER's Degree in DATA SCIENCE AND ENGINEERING

MASTER's Degree Thesis

Associative spatio-temporal classification: a scalable spatio-temporal classifier

Supervisors

Prof. Paolo GARZA

Dr. Luca COLOMBA

Candidate

Lorenzo OTTINO

July 2022

Abstract

The analysis of data characterized by spatial and temporal information of events presents several challenges. These challenges are related to finding meaningful representations of the data and being able to process them in massive quantities. When dealing with classification of spatio-temporal data, another open issue is the capability of identifying and predicting rare critical events. The objective of this thesis is to present a novel associative classifier to tackle all these previously mentioned problems.

The classifier was trained and tested on a public bike sharing dataset of the city of San Francisco Bay Area in a two-years long period. The model was built with Python and Spark. Moreover, we compared the performances of the proposed approach with three different classifiers. The results show superior performances in terms of precision, and a better resilience to missing values. The results highlight the importance of an effective data representation of the spatio-temporal events, and that interpretable models can provide good insights to enhance the performance of prediction algorithms, even in cases where the interpretability has not a crucial importance.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Bike sharing is a green and sustainable mobility solution. It is a very common and widespread transportation system nowadays. The principle of bike sharing consists in the possibility of using high numbers of bicycles located in a given area, usually a city. Bike sharing systems can be classified into station-based and free-floating services, depending on the flexibility of the implemented solution. The latter allows users to drop a bike at any location, whereas the first requires the users to drop the rented bike at the designated stations.
The most important qualities of this kind of transportation are its sustainability, contributing to the reduction of air pollution, and the health benefits it gives to the users if they include it in their lifestyle. In addition, it is a good solution to reduce the number of cars in cities, decreasing the problems in the transportation related to traffic.
The typical usage of this kind of mobility is for short trips. For this reason bike sharing has the majority of its applications in big cities and urban areas.

1.2 Goals

The main goal of this thesis is to assess whether an associative classifier is a suitable solution to tackle the analysis of data characterized by spatial and temporal information, and to predict the advent of rare events. In our context, the goal of the classification is to predict critical events, such as if a station is going to be empty or full in the next minutes. This kind of information has a crucial importance, because it can lead to the unavailability of the system to users. Managers of the bike sharing system may use such information to dynamically balance the number of bicycles in each station, and potentially to optimize the number of interventions

on the system. On a customer's perspective, this can lead to a better quality of service, but managers may also benefit of the extracted knowledge to reduce the expenses in the balancing operations.
The main advantages of an associative classifier are its interpretability and its capability to give insights into the relations between the events that occur in the analysed data. This work was structured into 4 phases:

1. Data exploration on the available dataset;
2. Transforming the data in suitable formats for the extraction of association rules and sequences;
3. Analysis of the results and training of the associative classifier;
4. Testing the classifier, also comparing its results with other applied classification models;

1.3 Structure

This work is structured in 5 main chapters:

- **Chapter 1**: introduction to the work
- **Chapter 2**: presentation of the necessary theory and concepts used in this thesis (Big Data, Association Rules, Classification)
- **Chapter 3**: detailed analysis of the available data
- **Chapter 4**: presentation of the experiments conducted with the associative classifier and the considered competitors, with their corresponding results
- **Chapter 5**: final analysis of the results, conclusions and possible future works

Chapter 2

Related Works

In this chapter, we discuss the main theoretical concepts and technologies that were used in this work. First, we define the concept of Big Data and introduce Apache Spark, the main framework adopted in this thesis. Then, we present what Association Rules are, and how we can extract them. Finally, we discuss what is classification, how association rules can be used for this purpose and the other classifiers used in order to compare the classification results.

2.1 Big Data

Several definitions of Big Data exist. In general, with these words we refer to data whose size can not be handled by normal computers, that are generated at a very fast rate, and have various and different formats [1]. Big data are typically characterized by the following 5 attributes:

- **Volume**: its size grows at very high rates over time;
- **Variety**: data is encoded in various formats, including numbers, images, videos, signals, plain text;
- **Velocity**: data is generated at high rates;
- **Veracity**: it contains information that can be exploited if correctly understood;
- **Value**: intrinsic economic value associated with the data itself;

The amount of data analyzed in this work did not necessarily require the use of architectures specifically designed for Big Data, however this approach was used for better generalization and extensibility to other domains. Moreover, this kind of datasets can easily reach high volumes.

In this context, we used the Apache Spark™ software together with the Python language. Apache Spark™ is an engine that allows the processing of big data volumes through a distributed architecture [2]. It supports the Python programming language.

2.2 Association rules

The extraction of Association Rules is a common exploratory technique in Data Science [3]. Given a dataset where each row contains a certain number of items, an association rule is a rule written in the format:

$$\langle itemA \rangle \Rightarrow \langle itemB \rangle \tag{2.1}$$

Where the $\Rightarrow$ symbol indicates co-occurrence. This means that when one or more of the items on the left are present, the items on the right are present with a certain probability. In this kind of representation, the items on the left side are called *body*, whereas the items on the right side are called *head* of the rule.

2.2.1 Definitions

Given a transactional database D, we need to provide the following definitions which are preparatory for the next chapters:

- **Transaction:** the set of items in a row of a database D;
- **Itemset:** set of one or more items;
- **Support:** fraction of transactions that contain an itemset. Given an itemset I, it corresponds to the ratio

$$Sup(I) = \frac{n}{T} \tag{2.2}$$

  where n is the number of transactions containing the itemset and T is the number of total transactions in the database D.

- **Confidence:** the frequency of an itemset A in all the transactions containing another itemset B, where A is in the head and B in the body.

$$Conf = \frac{Sup(A \bigcup B)}{Sup(B)} \tag{2.3}$$

2.2.2 Extraction

Given a database of transactions, the task consists in extracting all the rules that have a higher support than a pre-defined minimum threshold. The choice of the minimum threshold needs to take into account a trade-off: a very low value will lead to very high or unfeasible computational costs, but a too high one may lead to loose rare but relevant itemsets. Mining the association rules is not a trivial problem, given the unfeasible computational cost of the generation of all the possible permutations $\mathcal{O}\left(wT2^d\right)$ [3], where T is the number of transactions, d the number of items and w is the maximum transaction length. The most common solutions to extract association rules are the *apriori* and the *FP-growth* [4].

2.3 Sequence mining

In some cases the transactions contain information about time. We may want to exploit this information to link each other the events that regard a specific situation [5].
A **sequence** is an ordered list of elements:

$$s = \langle e_1 e_2 e_3 ... \rangle \tag{2.4}$$

Every **element** is made of a series of events:

$$e_i = \{i_1, i_2, ..., i_k\} \tag{2.5}$$

A **subsequence** is a sequence containing a subset of the elements present in another sequence with the same temporal order between elements.
The **support** of a subsequence is represented by the number of sequences that contain a specific subsequence.
We call **sequence mining** the extraction of all the subsequences above a given minimum support threshold.

2.3.1 Extraction

To tackle this problem we used the PrefixSpan [6] algorithm in its implementation in the Spark MLlib library. The main idea of this algorithm is, instead of considering all the possible occurrences of the frequent sub-sequences, to consider only the sequences that can be obtained from a frequent prefix [6]. Given a sequence $\alpha = \langle e_1, e_2, ...e_n \rangle$, a sequence $\beta = \langle e'_1, e'_2, ...e'_m \rangle$, $(m \leq n)$ is called a prefix of α if and only if (from [6]):

- $e'_i = e_i, \forall i \leq m - 1$

- $e'_m \subseteq e_m$
- all the items in $(e_m - e'_m)$ are alphabetically ordered after those in e'_m

This can be done because all frequent sub-sequences can be found starting from a frequent prefix.

2.4 Classification

The Classification task is a very well known supervised learning problem in the Machine Learning field. Its goal, given a vector of features X, and a qualitative response Y, is to build a function that takes X as input and predicts a value for Y [7]. The classification is called binary if its objective is predicting the presence of absence of a given attribute. Usually, in this kind of operations the dataset is divided into 2 portions: one, generally of bigger size, is called train set and is used for building the classifier. The second one, called test set, is used to evaluate the performance of the model. In the next sections we are going to introduce the classifiers, also called predictors, used in this work and the metrics adopted for assessing their performance.

2.4.1 Associative classifier

An associative classifier predicts the class labels according to previously extracted rules, where the rules are expressed in the format [8]:

$$\langle X \rangle \Rightarrow y \tag{2.6}$$

where the condition is the body of the rule and y is the class label. The body is made of a set of conditions *feature = value*. The association rules can be extracted in different ways as discussed in 2.2. Once we have extracted the association rules, we iterate over the test set and check if the test data satisfy the condition. In the positive case, we say that the rule matches and the predicted class is y. Some variants of this kind of classifier may require more than a rule to match to make a positive prediction. If no rules match, the default class will be the result of the prediction. The rules list is usually ordered. In our case they are ordered by decreasing confidence.

2.4.2 Decision Tree

"A Decision Tree is a predictor that predicts the label associated with an instance X by traveling from a root node of a tree to a leaf" [9]. In the case of Binary Classification, at each node on the path from the root to a leaf, the successor child is usually chosen based on a splitting of the domain of one of the features of X. Each leaf contains a specific label.

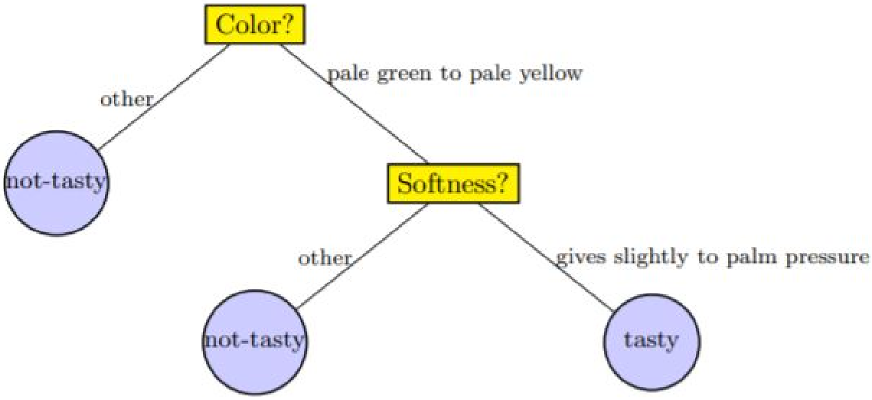

Figure 2.1: Example of a decision tree, from [9]

Practical decision tree training algorithms are based on heuristics such as a greedy approach. In this case, the tree is constructed gradually. According to the greedy approach, locally optimal decisions are made based on some splitting criterion [9].
The most common splitting criteria are:

$$GINI = \sum_{k=1}^{K} p_{mk}(1 - p_{mk}) \tag{2.7}$$

$$ENTROPY = -\sum_{k=1}^{K} p_{mk} log(p_{mk}) \tag{2.8}$$

Where K is the number of classes and p_{mk} represents the proportion of training observations in the m^{th} region that belong to the k^{th} class.

2.4.3 Random Forest

Random Forest is a predictor that builds several Decision Trees on bootstrapped training samples [10]. Bootstrap aggregation, or bagging, consists of taking repeated samples from the training set, generating many different bootstrapped datasets. This is usually done for reducing the variance of a statistical learning method. When building the Decision Trees, each time a split in a tree is considered, a

random selection of m features is chosen to split candidates from the full set of p features. The split is allowed to use only one of those features. A common choice is taking $m = \sqrt{p}$.
When performing a prediction in classification, Random Forest combines the predictions of all the obtained trees, usually with majority voting, meaning that the final prediction will correspond to the class predicted by the majority of the single trees.

2.4.4 XGBoost

Extreme Gradient Boosting, or XGBoost [11], is a tool based on the Boosting principle. The difference between Bagging and Boosting consists in the fact that the former takes bootstrapped training sets to build the predictors, whereas the latter builds the predictors according to the training data that were not correctly classified in the previous training rounds.

2.4.5 Evaluation metrics

Several measures of the quality of a classifier exist. Before introducing those used in this thesis, we need to present some definitions. Considering a Binary Classification problem, data points predicted by a classifier can be categorized into the 4 following cases:

- **True Positive (TP):** the predicted class is Positive and the prediction is correct;
- **True Negative (TN):** the predicted class is Negative and the prediction is correct;
- **False Positive (FP):** the predicted class is Positive and the prediction is wrong;
- **False Negative (FN):** the predicted class is Negative and the prediction is wrong;

A common representation of these values is with a confusion matrix (Figure 2.2). Given the confusion matrix, we can define the following evaluation metrics: accuracy, precision, recall and f1-score.

Accuracy

The first metric considered is the accuracy, that is defined as the number of correctly classified samples over the total size of the test set. A drawback of this

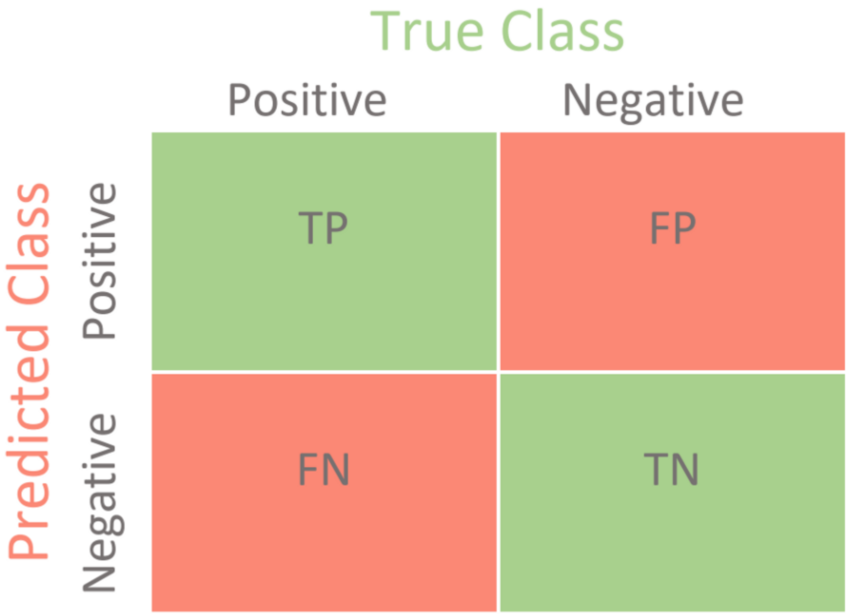

Figure 2.2: Example of a confusion matrix

metric is that it is not able to provide a good evaluation in presence of high class imbalance. We call class imbalance a situation where the number of actual positives is not comparable to the number of actual negatives (e.g. 70% positives and 30% negatives).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.9}$$

Precision

The aim of the precision is to identify the ability of a model in identifying the target class. It is calculated as the number of correct positive predictions over the total number of positive predictions.

$$Precision = \frac{TP}{TP + FP} \tag{2.10}$$

Recall

The goal of the recall is to assess the capability of the model in recognizing the positive class. It is calculated as the ratio of the correct positive predictions and the total positive test samples.

$$Recall = \frac{TP}{TP + FN} \tag{2.11}$$

F1 score

The F1 score is an the harmonic mean of the values of Precision and Recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.12}$$

Chapter 3

Analysis of the spatio-temporal data

In this chapter, we provide details about the considered datasets. Then, we present the algorithm for the transactions generation and the extraction of frequent sequences. Next, we present how they are used for the associative classification. Finally, we present the considered alternatives to make a performance comparison.

3.1 General overview

The first step was analyzing the datasets considered [12].The dataset contains data collected over 2 years in the bike sharing system of the San Francisco Bay Area, covering 5 different cities: San Francisco, San Jose, Palo Alto, Mountain View and Redwood City. The available data are related to bike sharing stations. The dataset provides the following information: geographical location and name of each station, maximal capacity, and the amount of bikes and docks available in each station, sampled at each minute during the 24 hours of the day. First, we performed some preliminary analysis. In particular, for each arrival station we considered the stations that have the highest number of bikes directed to the considered arrival station, i.e. the top departure station. For each tuple departure-arrival station, we computed the correlation between the number of bikes. Then, a final evaluation was performed to assess the consistency of the available dataset and the impact of eventual re-balancing operations for what concerns the number of bikes.
After this initial phase, the algorithm for the event generation and pattern extraction was implemented. We developed a Python class in order to consider different possible events of interest and vary several different parameters, such as the definition of neighbourhood and the thresholds for defining an event.
Then, the extracted patterns were used for an associative classification pipeline,

whose objective was to predict the emergence of critical events, corresponding to a station being almost full or empty. Finally, other classifiers were tested with the aim of making a comparison of the performance. The considered models are the Decision Tree, Random Forest and XGBoost. Different types of preprocessing, one dependent on the specific station id, and one independent, were considered for the input data in the other classifiers. The performance of our associative classifier are described in the following chapter and compared with other classification algorithms

3.2 Analysis of the dataset

The analysed data were organized in 3 files:

- Status: this file contains rows with the number of available docks and bikes for a station, identified by an id, together with a timestamp.
- Trips: this file contains data about all the recorded bike trips. Each row contains information about the start timestamp, end timestamp, departure station, arrival station, and the duration of the trip.
- Stations: this file contains the detailed information of each station, including its name, location, and maximal capacity.

3.2.1 Status analysis

Dataset reliability

The data availability of the status file ranges over a 2-year period (29/08/2013 - 31/8/2015). However, there is no guarantee that the status information is available for every station starting from 29/08/2013. Analysing the 70 existing stations, we discovered that most of them (64/70) cover the complete period. The remaining stations have the first record in a later day, probably because of the deferred installation of the station itself.
This dataset does not contain any missing or null values, however, even if the sampling frequency appears to be 1 minute there may be higher gaps between two consecutive records.
Another relevant observation should be done about this file: the sum of the available docks and the available bikes in each station should always be constant and coherent with the nominal value present in the "Stations" file. However, this number presents oscillations in the period, probably because of little failures in the data gathering system. The records where the modulus of the difference between the nominal value and the calculated one is greater than a threshold = 5 have been discarded because not considered reliable.

Events analysis

Later, the occurrence of all the critical events was studied. We can define the following critical events or states:

- Full: the total number of docks available is equal to 0;
- Almost-Full: the total number of docks available is ≤ 2;
- Empty: the total number of bikes available is equal to 0;
- Almost-Empty: the total number of bikes available is ≤ 2;
- Normal: all other cases;

The critical events distribution was calculated for some representative stations. Results (Figures 3.1, 3.2, 3.3) show that in a single day the station is mostly in the "Normal" state, and that critical events such as Empty or Full happen rarely.

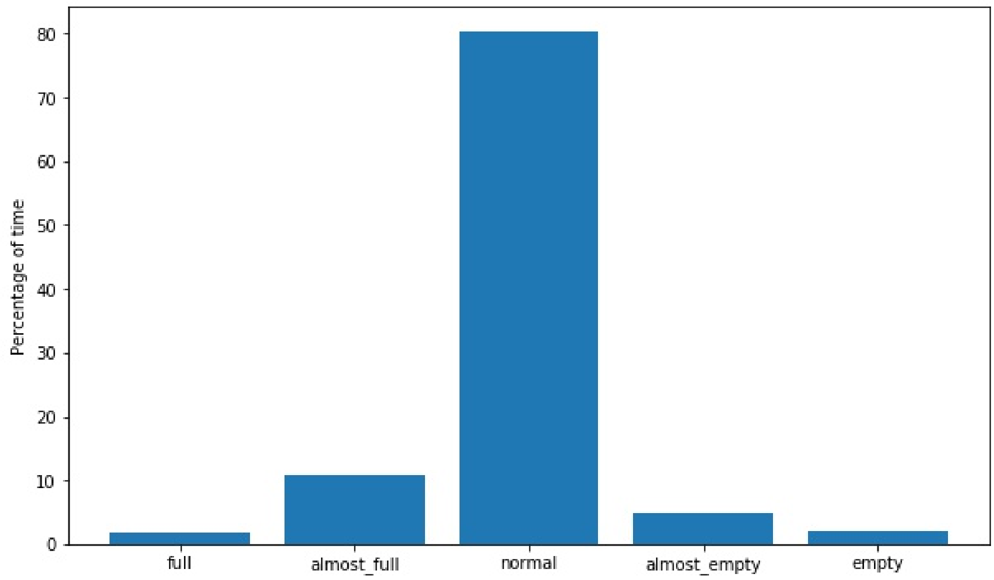

Figure 3.1: Average time spent by Station 70 (San Francisco, Caltrain) in each considered state

3.2.2 Trips analysis

Trips duration

This file stores all the anonymized bike trips in the aforementioned 2-year time period. Initially, the distribution of the trips duration was investigated. Even if

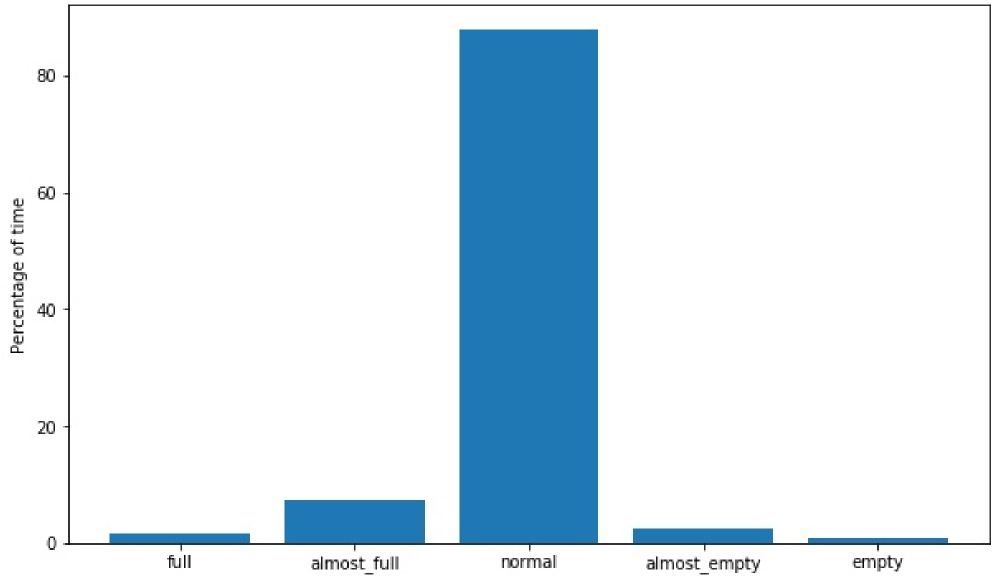

Figure 3.2: Average time spent by Station 72 (San Francisco, Center BART) in each considered state

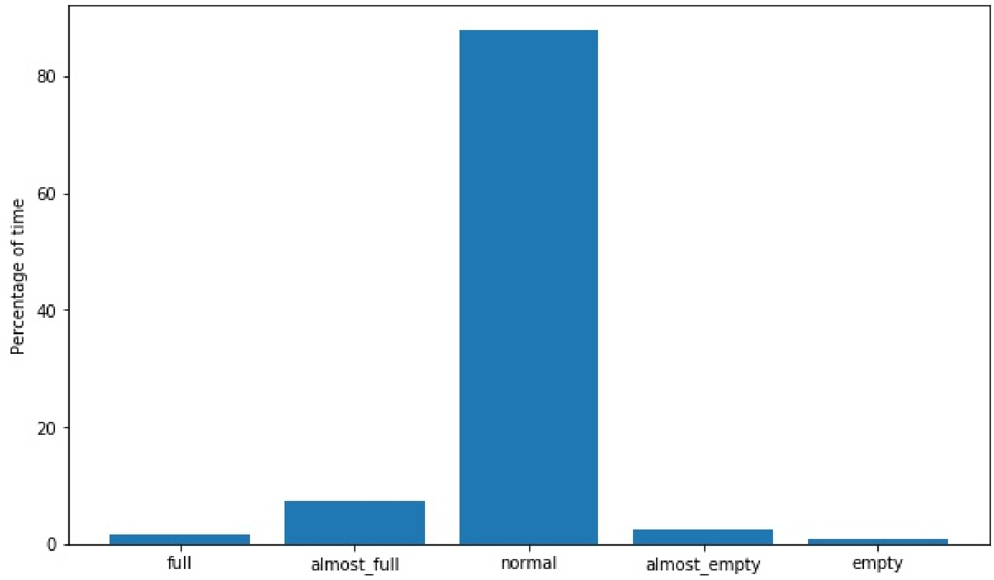

Figure 3.3: Average time spent by Station 34 (Palo Alto, Caltrain Station) in each considered state

there are some outliers, most of the trips have a duration of between 5 and 20 minutes. Figure 3.4 shows the box plot of the trips duration after the outliers removal. The trips duration was also investigated for the two different groups of possible users (Customers and Subscribers). The results show that the majority of the users (the subscribers class) takes on average shorter trips than the occasional users.

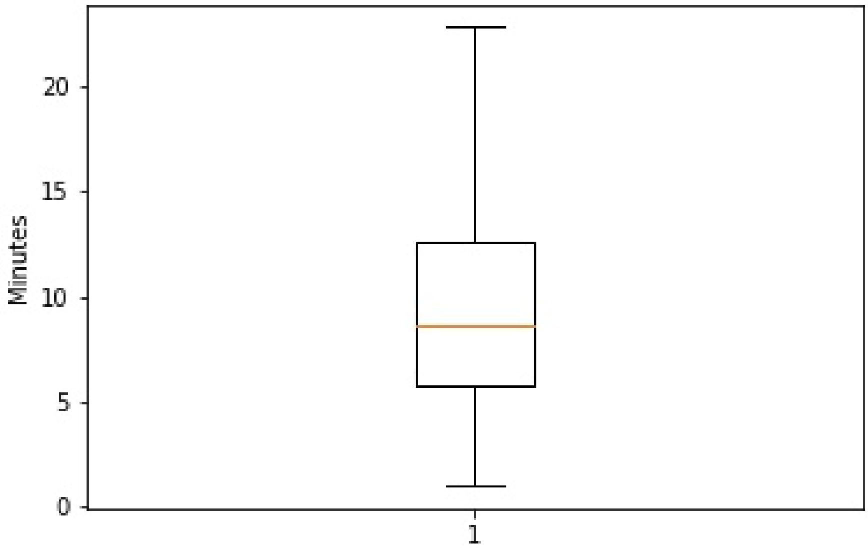

Figure 3.4: Average time spent by a user of the bike sharing system on a trip

Graph view of the system

It is clear that this system can be seen as a graph, where the stations are the nodes and the trips from a station to another represent the links. We may see it both as a directed or undirected graph, and we can define the weight of each link as the total number of trips on that link in the considered period.
As we might expect, the graph is connected, but not complete, and the weights values are not equally distributed, pointing out that some stations and links are more used than others. Following this observation, the connected components on the undirected graph have been analysed after removing the links whose weights are below a threshold close to the first quartile. The graph has 5 connected components corresponding to the 5 cities. This means that the bike sharing system is mostly used within the same city, even if trips from one city to another are present.
We can also analyze the directed graph: in this case the number of incoming and outgoing bicycles, except for some special cases, share similar values. These values

should ideally be equal, however this is not the case because of the re-balancing operations that are normally performed on a bike sharing system to tackle the problem of having completely full or empty stations. In this analysis, the number of incoming bicycles has been defined as the sum of the weights of the incoming links, whereas the number of outgoing bicycles has been defined as the sum of the weights of the outgoing links. All the stations have self-loops, but it is a minority of the trips in all cases.

Usage analysis

Another relevant information regards the usage of the bikes. All the trips were grouped according to their start time, and all the instances were counted. The result in Figure 3.5 shows that the bike sharing system is mostly used in day time, with two peaks of usage in correspondence of the beginning and the end of the working day.

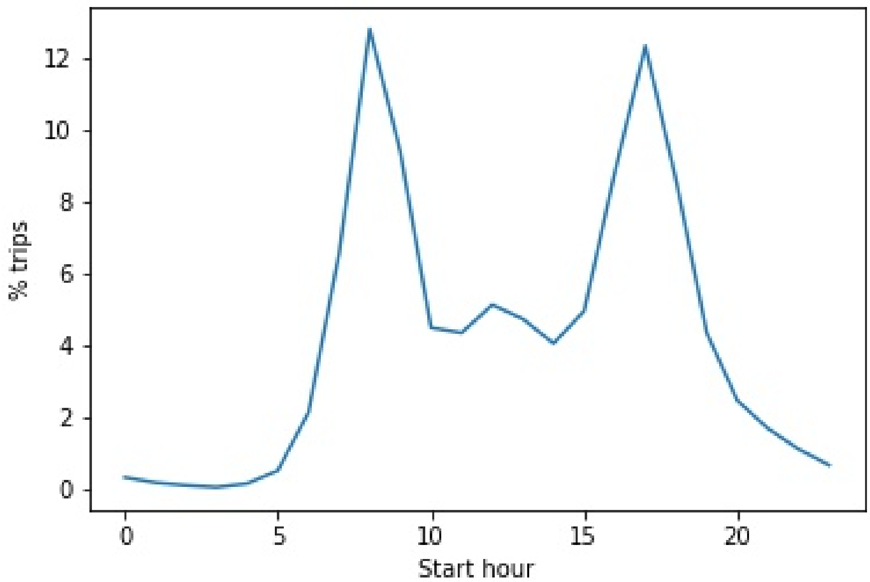

Figure 3.5: Percentage of the trips for each hour of the day

Consistency analysis

A final consideration was done regarding the consistency of the data available in the "Status" and "Trips" files. Starting from the latter, reconstructing the former should be theoretically possible by counting the bikes arrived and departed, given

the initial offset of the status of all the stations.
By doing so, we discover that the number of bikes in a station always tends to $-\infty$ or $+\infty$ (Figures 3.6, 3.7).
If we assume that the measurements do not contain enough errors to compromise the reliability of the dataset, this different result is due to the re-balancing operations. These operations appear to be applied in little amounts (no more than 10 units) during the day, when critical events occur. We can see an example of this in Figure 3.8, where the status is reconstructed for a single day only.

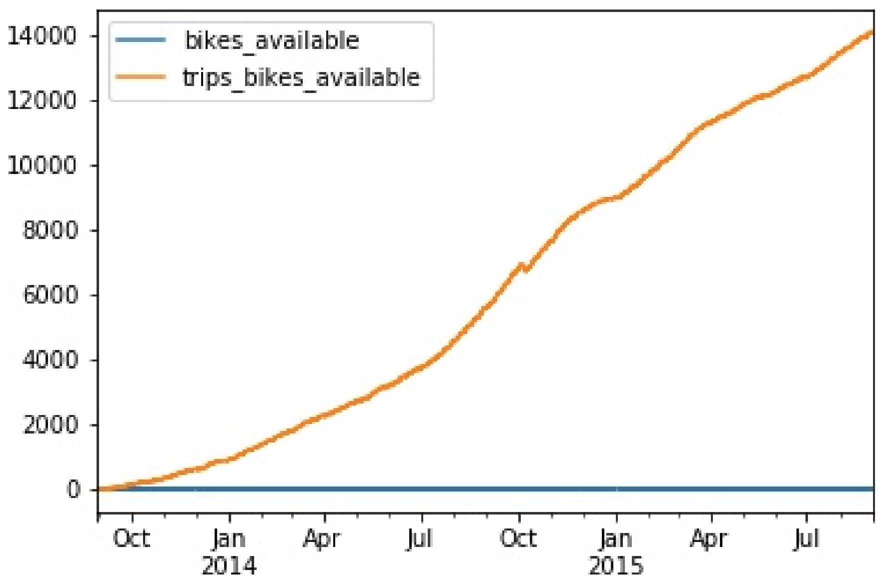

Figure 3.6: Status of station 70, reconstructed from the trips data for all the time recorded in the dataset.

3.2.3 Correlations

The goal of this analysis was to establish how much the number of bikes in a single station is related to the number of bikes in its neighbours. For each single station, the neighbourhood was defined as the set of all the stations that have an outgoing or incoming link to the selected station, regardless the distance. The reason for this choice is that trips are not, in most cases, between two adjacent stations. First, we defined a time window of X minutes. For each station within non-overlapping time windows, we computed the average number of bicycles. Then, for each station we considered 5 consecutive time windows with their corresponding values to compute

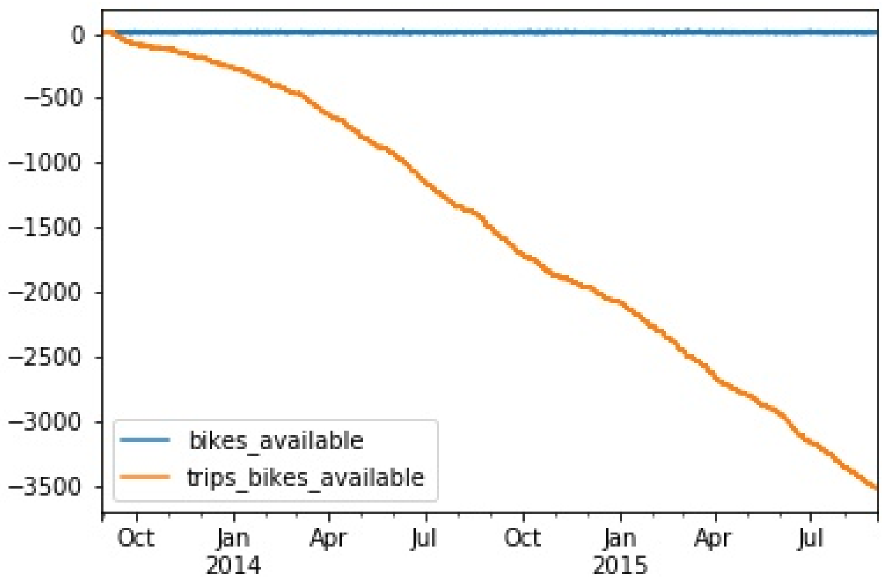

Figure 3.7: Status of station 56 (Beale at Market, San Francisco), reconstructed from the trips data for all the time recorded in the dataset.

their correlation between all the neighboring stations. Several values of the intervals were tested.
A further analysis was conducted after clustering the data in time slots, since the usage of the bike sharing system is not constant along the day. The time slot values were chosen according to the analysis in Section 3.2.2.
Some results are shown in the Figures 3.9, 3.10, 3.11, 3.12. In general, the number of bikes is highly correlated to its value in the preceding intervals, and does shows only weak relations with the other stations' values.

3.2.4 Bike variations analysis

A final observation was done on the variation of the number of bikes in each station. After defining an interval, we calculate the average number of bikes in that interval, then we calculate the difference with the preceding interval. In this way we obtain an indicator of the amount of the variations that usually occur (Figure 3.13).

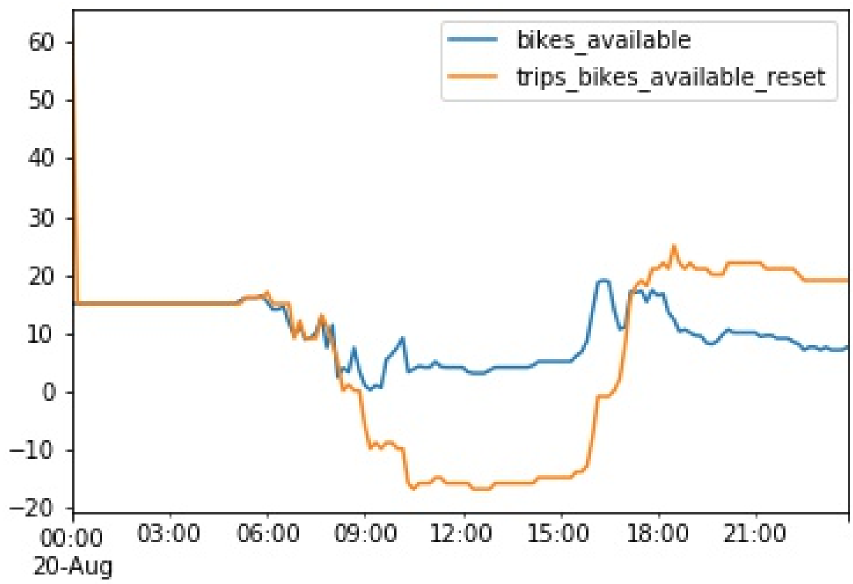

Figure 3.8: Status of station 70, reconstructed from the trips data for a single day with daily reset.

3.3 Pattern extraction

The purpose of this section is to present the algorithm for processing the spatio-temporal data, together with the considered parameters and different options.

3.3.1 Discrete event generation algorithm

The available data are specific for each station and timestamp. The goal of this extraction is to produce items that do not refer to specific stations or timestamps, but that are general and have the same structure for any station.
In this process, we will use discrete representations of time and space to encode the time intervals and the distance between two stations. Given a time threshold t, we first discretize the temporal axis with non-overlapping time windows of width t. Thus, we determine for every reading of the considered dataset its corresponding time window. Consequently, multiple readings of the considered dataset belong to the same discretised timestamp/time interval. Similarly, we apply spatial discretization based on distances between two different stations. Given a spatial threshold s and a reference station, the reference station is at distance 0 from itself, whereas all the other stations whose distance is between 1 and s are at distance 1

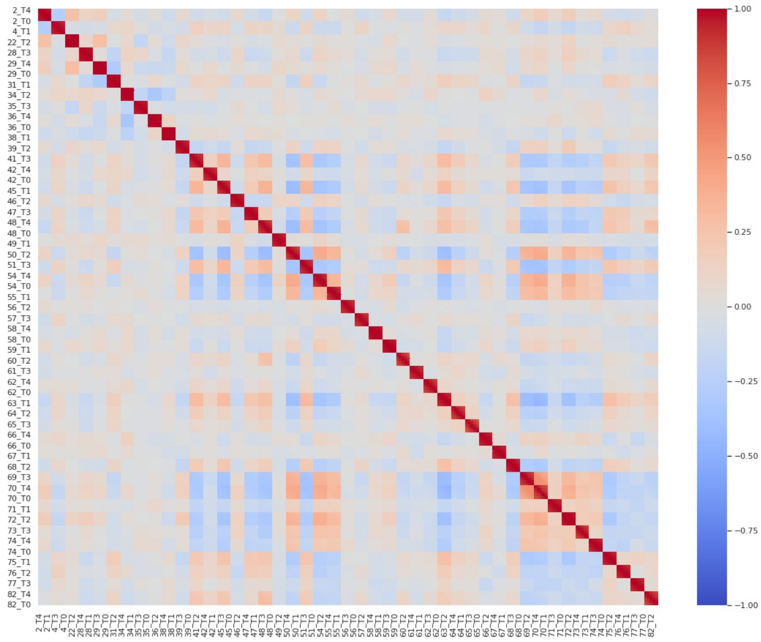

Figure 3.9: Correlation matrix for station 70 and its neighbors and intervals of 10 minutes.

from the reference, and so on.
The result of the event discretization will be as follows: for each discretized timestamp (marked as T0, T1, ...), and each station we will extract all the relevant states for the considered station and its neighbor stations. Then, a sequence of events is generated for each station in each discretized timestamp, considering all the events occurring in the station of interest and its neighbors (or likely departing station). An example of the final result, if we consider only the "Empty" state (meaning that there are no bikes available in the considered station) and stations at maximum distance equal to 1 in 3 time intervals, the Station A in Figure 3.14 would generate:

$$\begin{aligned}
&[[Empty_T0_0, Empty_T0_1],\\
&[Empty_T1_0, Empty_T1_1],\\
&[Empty_T2_0]]
\end{aligned}$$

This sequence of items is extracted if the considered station and at least a station at distance 1 are empty for two consecutive intervals and in the subsequent

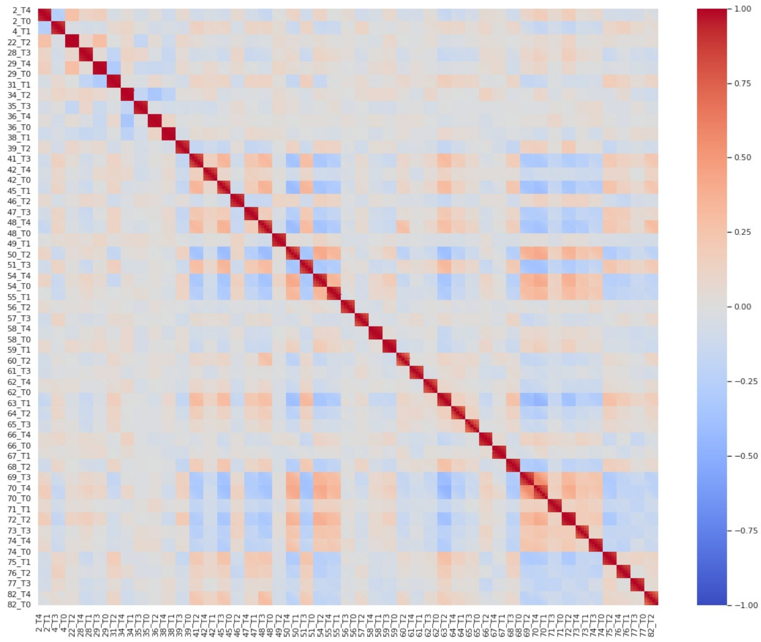

Figure 3.10: Correlation matrix for station 70 and its neighbors and intervals of 20 minutes.

interval no stations at distance 1 contain 0 bikes.
This extraction algorithm was implemented in a PySpark application.

3.3.2 Algorithm parameters

The parameters considered in the proposed methodology are:

- Extraction type: the type of events considered as relevant, e.g. Full, Empty, etc.

- Neighborhood type: the neighbor stations considered may be according to their distance to the considered station ("distance") or according to the weight of the incoming or outgoing links ("indegree"). In this case, only the top X stations are selected, where X is another parameter. In the former case, only the stations at a discrete distance $\leq Z$ are considered, where Z is another parameter;

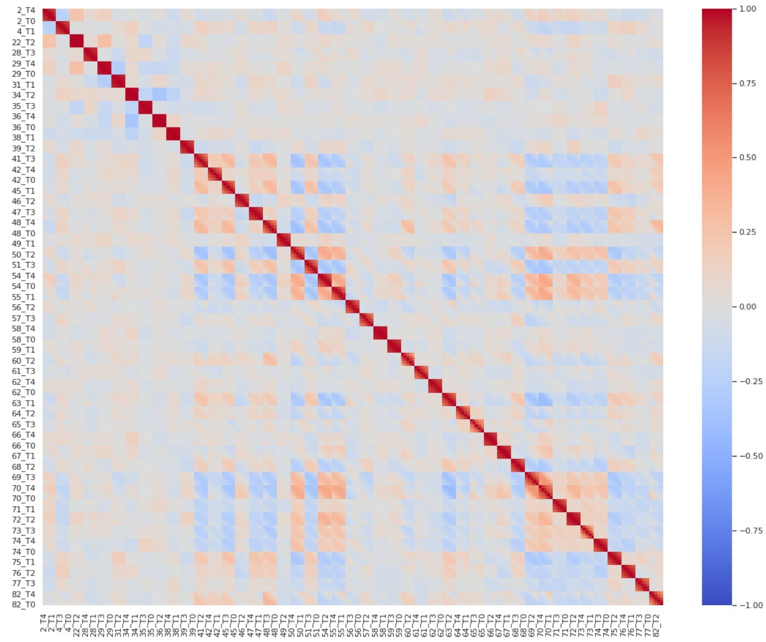

Figure 3.11: Correlation matrix for station 70 and its neighbors and intervals of 20 minutes for the time slot 5:00 - 9:59.

- Threshold for increase/decrease states (ID-TH): if we consider the "increase" or "decrease" states, this threshold corresponds to the minimum variation in the number of bikes in the number of bikes in a station within a time interval to generate the "increase"/"decrease" event;
- Extraction target: Full or Empty, it indicates the objective of the event generation for the reference station;
- Threshold for "almost critical" state (AC-TH): this parameter defines the maximum value of available bikes/docks to consider a station almost full or almost empty, according to the specified target;
- Wrap states: boolean parameter. If true, consider the states "almost critical" and "critical" as the same state, if false, the two events are considered separately;
- State change: boolean parameter. If true, consider the existence of a critical state only if in the first timestamp of the time interval it was not critical;

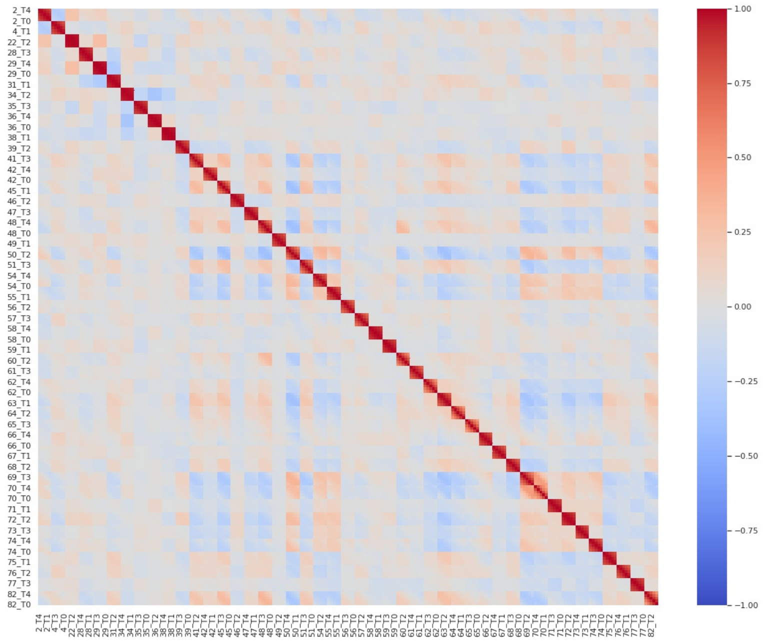

Figure 3.12: Correlation matrix for station 70 and its neighbors and intervals of 20 minutes for the time slot 15:00 - 20:59.

- Time Window (TW): width of the time window for time discretization (in minutes);
- Window size: how many consecutive windows are considered when generating sequences;
- Time zone: consider only some given hours inside the day, e.g. only the events occurred between 8 and 12 am;
- Negative increase/decrease: boolean parameters. If true, map the state "not increasing" and "not decreasing";
- #matches: when performing the classification, indicates the minimum number of rules that need to match for a positive prediction;
- min confidence: when performing the classification, indicates the minimum threshold for actually using the rule in the set of rules for a prediction;

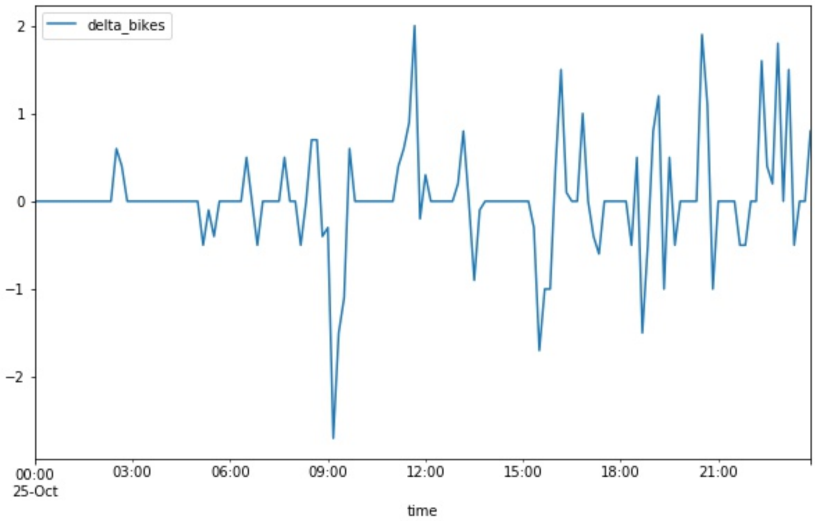

Figure 3.13: Example of the variation of the number of bikes in a day for station 70 with interval of 10 minutes.

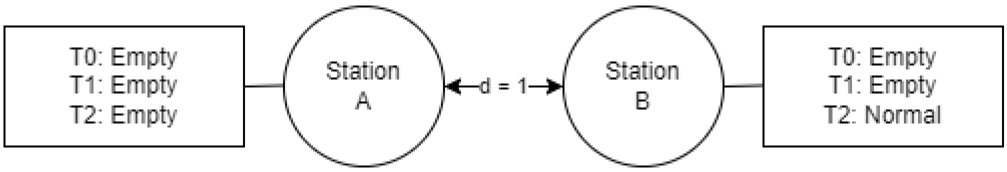

Figure 3.14: Example of a situation from which we can generate the items

3.3.3 Algorithm steps

Here we explain the general steps of the algorithm:

1. load "Status" data. The input data format is (stationID, #bikes, #docks, time). For sake of simplicity, we will consider the time in Unix format (number of seconds from 01/01/1970);
2. filter the rows that contain null or meaningless data;
3. group rows by Station and WindowID. The WindowID is an integer number obtained with the formula $W_{ID} = math.floor(\frac{time}{interval*60})$, where the interval is in minutes. Then for each relevant state map a row (stationID, windowID, state);
4. group all the couples (stationID, state) by their windowID;

5. obtain for each windowID the data associated with preceding windowIDs (2 in our example case, the total must be equal to the number of considered time intervals);

6. for each station in the window, map the obtained data to relative distances to the considered station;

7. remove the data about stations not belonging to the neighborhood of the considered station, then remove the label about the specific station and windowID;

3.4 Associative classification

This section is devoted to discussing how the classification process is handled for the associative classifier. For this and all the experiments performed, the "Status" dataset was divided in the same 70/30 train/test split.

3.4.1 Frequent patterns extraction

After generating the events, the most frequent patterns are extracted with the Prefix Span algorithm. Then, the patterns are filtered by a confidence threshold, and only those that contain the target state are kept. We refer to the target state as the item with temporal distance 0 and spatial distance 0, i.e., T0_S0, the reference station at the first considered discretized timestamp. These are the final association rules.
Then the items are generated from the test set, in the same way as before, but considering also the "Normal" state, corresponding to the absence of the target critical event. For all the generated objects, the rules are compared. In this comparison, each single item in a rule is compared with the items in the test object. If all the items in the body of a rule are present, we say that the rule matches with the object. The number of rules that need to match for making a prediction needs to be specified as a parameter. If enough rules match with the item, the final state will be classified as critical, normal otherwise. The workflow is shown in Figure 3.15. For all these experiments, a baseline was also implemented. This baseline will be referred to as "Dummy classifier" in the next chapters, and consists in predicting as the future state the current state. For instance, if a station is currently in "Normal" state, it will be predicted as "Normal" also in the next interval.

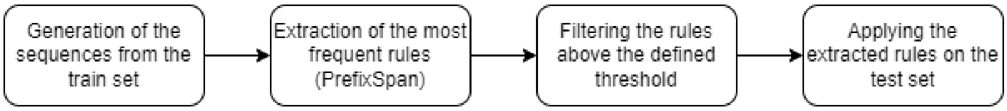

Figure 3.15: Block diagram of the phases regarding the training and the application of the associative classifier

3.5 Other approaches

In this section we discuss the other approaches adopted to tackle the problem and for comparing the results. Two different strategies for the data manipulation were evaluated. For each strategy, a Decision Tree, a Random Forest, and a XGBoost classifier were tuned maximizing their recall, precision and f1 score on the train set and evaluated on the test set. The same train/test split described in Section 3.4 was used for these experiments. We will now describe the two different data manipulation strategies adopted.

3.5.1 Station-specific approach

This approach requires to build a dataset and a classifier specifically for each station. After defining a time interval, we calculate the average number of docks available in that interval for each station and all the neighboring stations at a distance lower than a defined parameter. Then, we generate a column for each station and each time interval containing the mean bikes/docks available. The label value will be calculated according to the average number of bikes in the considered station in the next time interval. A drawback of this approach is that a missing value in a specific station will cause the deletion of the entire row, thus loosing some potentially useful data. For example, if we consider the toy dataset in Table 3.1, a time interval of 30 minutes and 2 consecutive intervals, the result will be 3 different datasets with one row, each one as in Table 3.2.

3.5.2 Station-non-specific approach

With this approach we follow a pre-processing step more similar to the one adopted with the associative classifier. Here, we build a unique common dataset for all the stations, hence a classifier for each station is not necessary. After defining the size of the time interval, the number of intervals to consider, and a maximum discrete distance D as in Section 3.4, we iterate over all the stations in the distance range between 0 and D. During this iteration, we check if at least one station is in one of the relevant states (e.g. "AlmostFull", "Increase" and "Decrease"): in the positive case, we mark with 1 the column associated to the corresponding

Timestamp	StationID	Bikes available	Docks available
15:01	1	0	15
15:31	1	1	14
16:01	1	1	14
15:01	2	1	14
15:31	2	1	14
16:01	2	5	10
15:01	3	7	8
15:31	3	7	8
16:01	3	8	7

Table 3.1: Example of training data

B T0_S1	B T1_S1	B T0_S2	B T1_S2	B T0_S3	B T1_S3	D T0_S1	D T1_S1	D T0_S2	D T1_S2	D T0_S3	D T1_S3	Label
0	1	1	1	7	7	15	14	14	14	8	8	AlmostFull
0	1	1	1	7	7	15	14	14	14	8	8	Normal
0	1	1	1	7	7	15	14	14	14	8	8	Normal

Table 3.2: Example of training dataset with station specific approach.

discretized temporal and spatial distance, indicating that at the given temporal and spatial distance there is at least one critical station. The target label will correspond to a boolean variable encoding the presence of the critical state in the next time interval for the station at distance 0 (the considered station). For example, if we consider the toy dataset in Table 3.1, the events "AlmostFull" and "Increase", a time interval of 30 minutes and 2 consecutive intervals, maximum distance as 2, and the distances between the stations are: [1-2: 1, 1-3: 2, 2-3: 1] the rows generated will be as in Table 3.3.

AF_T0_0	**I_T0_0**	**AF_T1_0**	**I_T1_0**	**AF_T0_1**	**I_T0_1**	**AF_T1_1**	**I_T1_1**	**AF_T0_2**	**I_T0_2**	**AF_T1_2**	**I_T1_2**	**Label**
1	Na	1	1	1	Na	1	0	0	Na	0	0	1
1	Na	1	0	1	Na	1	1	0	Na	0	0	0
0	Na	0	0	1	Na	1	0	1	Na	1	1	0

Table 3.3: Example of training dataset with station non-specific approach.

Chapter 4

Experiments

This chapter is devoted to presenting the experimental part of this thesis and its results. First, we present with more details the different configurations considered for the event generation, and we discuss the results. Then, the classification experiments targeting the "AlmostFull" and the "AlmostEmpty" cases are described, and finally we present the results of the other considered approaches.

4.1 Pattern Extraction

The event generation algorithm adopted for this and all the following experiments is the one described in Section 3.3.1. The number of patterns, and their structure varies according to the specific experiment run. In the following sections we will present all the necessary details about the parameters used and the impact they have on the final result. For the evaluation of the extracted rules we will consider their confidence and support values.

4.1.1 Full-AlmostFull

This is the first experiment considered. With this name, we mean that the parameters were set in order to generate only the events "Full" and "AlmostFull". The experiments were run both for "distance" and "indegree" neighborhood definitions, and the time window values considered are 15 and 30 minutes. For this experiment these parameters were used:

- minimum support for PrefixSpan: 0.05;
- discrete distance unit: 500 meters;
- maximum discrete distance in the "distance" case: 3;
- number of likely departing stations in the "indegree" case: 20;

Neighborhood: Distance, Time Window: 15

In this experiment, there are 225 extracted patterns in total. Their confidence distribution (Figure 4.1) shows that the rules are mostly distributed in the 0.7 - 0.95 range. Only 62 patterns contain the reference station in the head of the rule. Among those, the rule with higher confidence is

$$['AlmostFull\_T0\_0, AlmostFull\_T0\_1'],$$
$$['AlmostFull\_T1\_0, AlmostFull\_T1\_1'],$$
$$['AlmostFull\_T2\_0']$$

and has a confidence of 0.860 and a support of 16876.

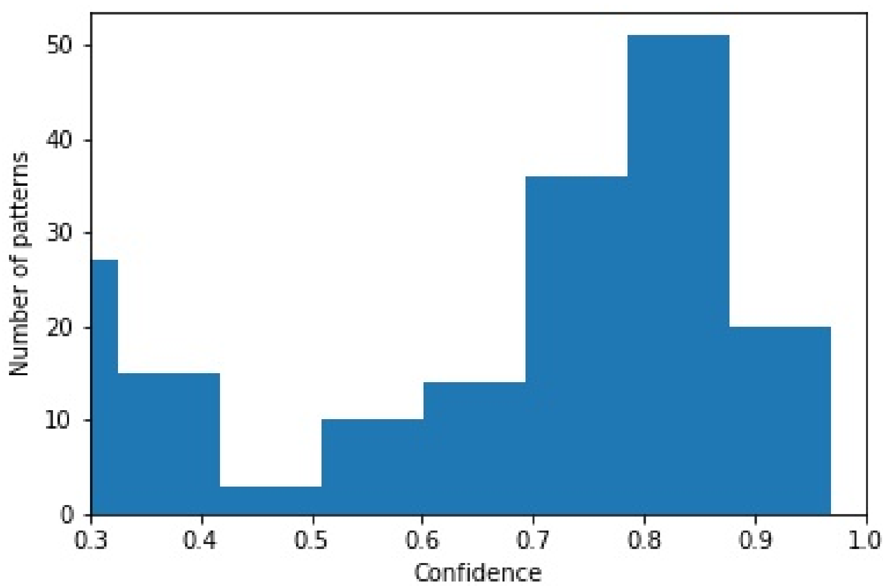

Figure 4.1: Confidence distribution of the patterns extracted in the experiment "Full-AlmostFull" with Neighborhood Distance and Time Window = 15

Neighborhood: Distance, Time Window: 30

In this experiment, there are 437 extracted patterns in total. Their confidence distribution (Figure 4.2) shows that the ranges with more patterns are between 0.6 and 0.85. Only 110 patterns contain the reference station in the head of the rule.

Among those, the rule with the highest confidence is

$$['AlmostFull\_T0\_0'],$$
$$['AlmostFull\_T1\_0'],$$
$$['AlmostFull\_T2\_0']$$

and has a confidence of 0.812 and a support of 70744. Such rule is also the third rule by support value. Furthermore, the rule with the highest support is

$$['AlmostFull\_T0\_0'],$$
$$['AlmostFull\_T1\_0']$$

with confidence of 0.799 and support of 87132. This points out the fact that in many cases the situation tends to be stable.

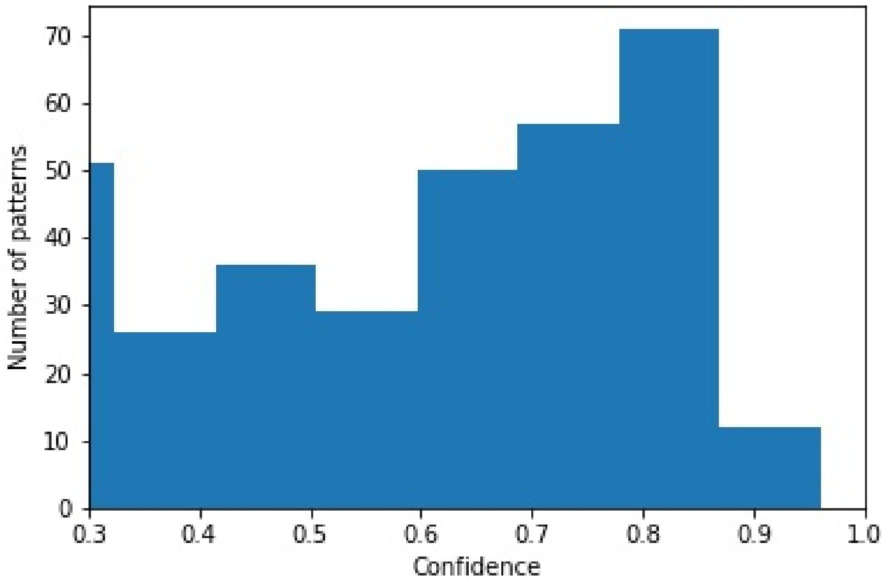

Figure 4.2: Confidence distribution of the patterns extracted in the experiment "Full-AlmostFull" with Neighborhood Distance and Time Window = 30.

Neighborhood: Indegree, Time Window: 15

In this experiment, there are 148 extracted patterns in total. Their confidence distribution (Figure 4.3) shows that almost all the patterns are in the range 0.55 -

0.9. Only 59 patterns contain the reference station in the head of the rule. Among those, the rule with the highest confidence is again

$$['AlmostFull\_T0\_0'],\\ ['AlmostFull\_T1\_0'],\\ ['AlmostFull\_T2\_0']$$

and has a confidence of 0.860 and a support of 135859.

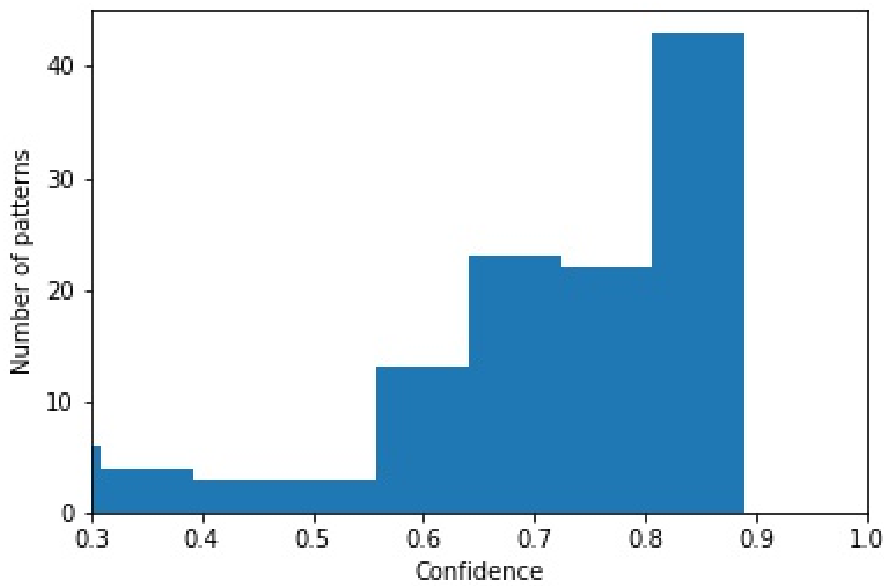

Figure 4.3: Confidence distribution of the patterns extracted in the experiment "Full-AlmostFull" with Neighborhood Indegree and Time Window = 15.

Neighborhood: Indegree, Time Window: 30

In this experiment, there are 274 extracted patterns in total. Their confidence distribution (Figure 4.4) shows two peaks in the number of patterns in range 0.6-0.7 range and near to 0.8. Only 86 patterns contain the reference station in the head of the rule. Among those, the rule with the highest confidence is once more

$$['AlmostFull\_T0\_0'],\\ ['AlmostFull\_T1\_0'],\\ ['AlmostFull\_T2\_0']$$

and has a confidence of 0.812 and a support of 70744, exactly as in Section 4.1.1.

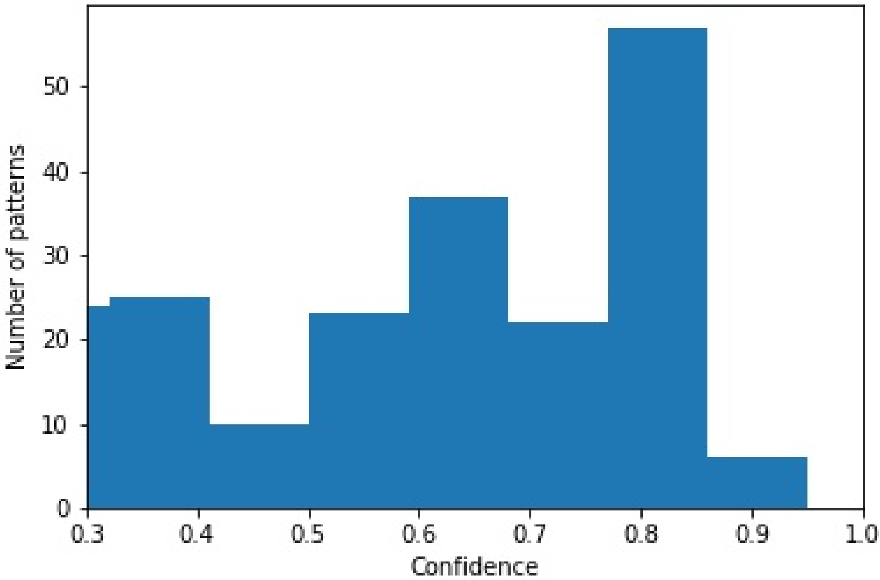

Figure 4.4: Confidence distribution of the patterns extracted in the experiment "Full-AlmostFull" with Neighborhood Indegree and Time Window = 30.

4.1.2 Full-Decrease

In these experiments, the events considered for the event generation are "Full", "AlmostFull" and "Increase" for the reference station, and "Decrease" for the neighbors. The purpose of this kind of extraction is to highlight the impact of "Decrease" events in departure stations on the reference station. The time window values considered are 15 and 30. Also the threshold for "increase" and "decrease" states and the minimum support for PrefixSpan values were changed. Different values for the number of likely departing stations were considered. For this experiment these fixed values were used for the parameters:

- Neighborhood type: "indegree";
- discrete distance unit: 500 meters;

Here all the tested configurations are listed:

1. #Neighbors: 10, ID-TH: 0, TW: 15, MS: 0.1
2. #Neighbors: 10, ID-TH: 1, TW: 15, MS: 0.1
3. #Neighbors: 10, ID-TH: 1, TW: 15, MS: 0.15

4. #Neighbors: 10, ID-TH: 2, TW: 15, MS: 0.15
5. #Neighbors: 20, ID-TH: 0, TW: 15, MS: 0.001
6. #Neighbors: 20, ID-TH: 0, TW: 15, MS: 0.1
7. #Neighbors: 20, ID-TH: 0, TW: 30, MS: 0.001
8. #Neighbors: 20, ID-TH: 1, TW: 15, MS: 0.001
9. #Neighbors: 20, ID-TH: 1, TW: 15, MS: 0.01
10. #Neighbors: 20, ID-TH: 1, TW: 15, MS: 0.1
11. #Neighbors: 20, ID-TH: 1, TW: 15, MS: 0.15
12. #Neighbors: 20, ID-TH: 1, TW: 30, MS: 0.001
13. #Neighbors: 20, ID-TH: 2, TW: 15, MS: 0.001
14. #Neighbors: 20, ID-TH: 2, TW: 15, MS: 0.15
15. #Neighbors: 20, ID-TH: 2, TW: 30, MS: 0.001

The experiments run have some commonalities, and we can identify some general effects in changing the parameters.

Changing the number of likely departing stations

Changing the number of potential departure station has obviously an impact on the number of the patterns extracted. As one might expect, selecting a higher number of neighbors allows to extract more patterns. The reason is surely related to the fact that generating more events allows considering more patterns. We can compare for example the experiments #2 and #10. For these configurations the only difference lies in the different number of likely departing stations (10 and 20). The number of extracted patterns in the former case is approximately 2/3 of the latter, having extracted respectively 17 and 29 patterns. However, the number of patterns containing the reference station in the head of the rule is exactly the same (11). The "missing" patterns have not the highest values of confidence, but some still have some interesting values, as we can see in the Figures 4.5 and 4.6. Also, the couples of experiments #1-#6 and #4-#14 share the same characteristics, with even higher reductions in the number of extracted patterns. In these cases, when the number of extracted patterns is higher, also the number of rules that contain the reference station in the head is decreased for the cases where less neighbors are considered. This reduction does not necessarily involve only lower-quality patterns, as we can see in the Figures 4.7 and 4.8.

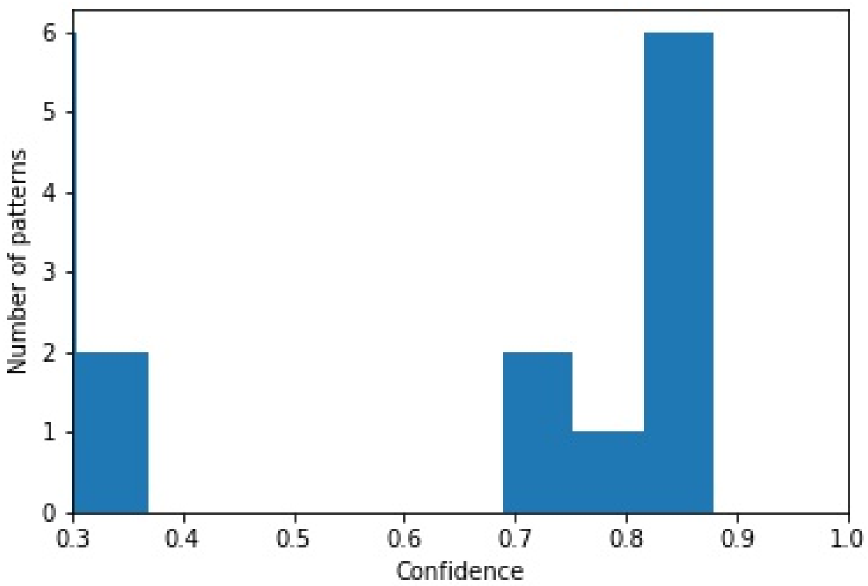

Figure 4.5: Confidence distribution of the patterns extracted in the experiment "Full-Decrease" #2.

Changing the Time Window

Also changing the time window value impacts the number of extracted patterns. As a consequence, also the confidence of the patterns seems to increase. The results are summarized in Table 4.1. It is difficult to infer why more patterns are extracted with larger windows. A possible explanation is that the difference lies in the data itself.

Experiment number	Patterns	Patterns with reference station
5	69272	23700
7	101991	29379
8	40320	12601
12	64464	22327
13	13491	3718
15	45142	13870

Table 4.1: Comparison of the pattern extracted with different time window values for the Full-Decrease experiment.

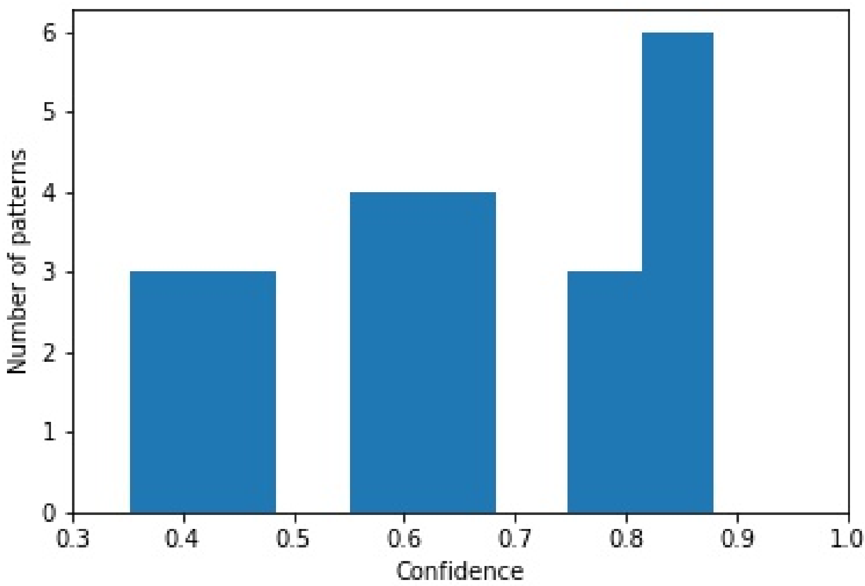

Figure 4.6: Confidence distribution of the patterns extracted in the experiment "Full-Decrease" #10.

Changing the ID-TH value

This is an important parameter, because changing the threshold defining the increasing or decreasing event affects the definition of the problem itself. Setting it equal to zero means that all the differences in the number of bikes are considered an increase/decrease event, meaning that we risk to be too much sensitive to small variations. On the contrary, too high values may not be able to spot differences that may impact other stations. Since the threshold for the "Almost Critical" event was set to 3, values higher than 2 were not tested.
We can see an example of this in the experiments #5, #8, and #13. In these cases, increasing the parameter's value leads to a huge decrease in the number of patterns. The respective values in fact are: 69272, 40320, and 13491. If we consider only the rules with the reference station in the head, the situation is not different: in this case the values are 23700, 12601, and 3718

Changing the minimum support for PrefixSpan

As previously discussed, changing this parameter surely affects the number of extracted patterns by PrefixSpan. Higher values lead to less patterns extracted, but also a faster computational time. It is a good parameter to decrease for more

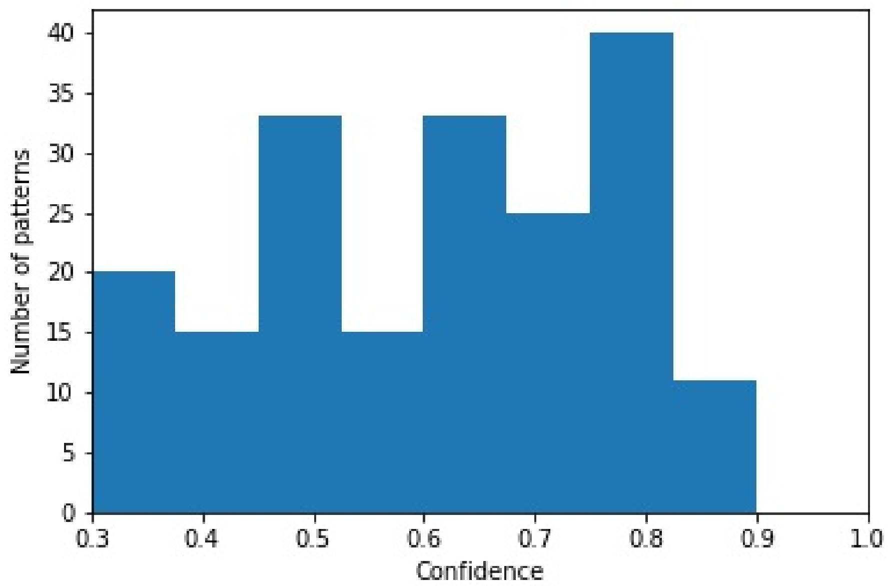

Figure 4.7: Confidence distribution of the patterns extracted in the experiment "Full-Decrease" #1.

accurate results, as long as the computational time remains acceptable.

4.1.3 Full-Decrease-stateChange

In these experiments, the events considered for the event generation are the same as the "Full-Decrease" experiment, but the definition of "Full" and "AlmostFull" states is slightly changed (see State Change parameter in Section 3.3.2). The time window values considered are 15 and 30. Also, the threshold for "increase" and "decrease" states values were changed. Different values for the number of likely departing stations were considered. For this experiment these fixed values were used for the parameters:

- Neighborhood type: "indegree";
- discrete distance unit: 500 meters;
- minimum support for PrefixSpan: 0.001;

Here all the tested configurations are listed:

1. #Neighbors: 10, ID-TH: 1, TW: 15

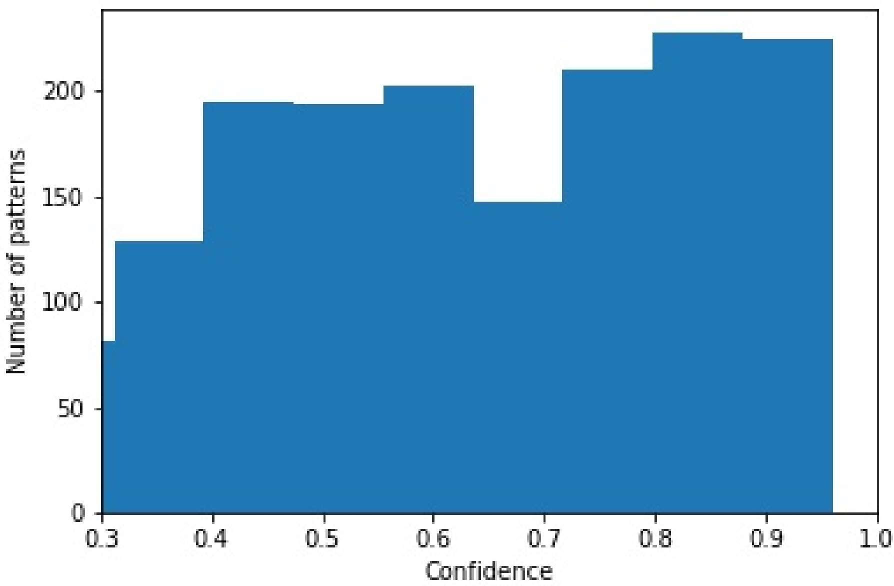

Figure 4.8: Confidence distribution of the patterns extracted in the experiment "Full-Decrease" #6.

2. #Neighbors: 10, ID-TH: 1, TW: 30
3. #Neighbors: 10, ID-TH: 2, TW: 15
4. #Neighbors: 10, ID-TH: 2, TW: 30
5. #Neighbors: 20, ID-TH: 1, TW: 15
6. #Neighbors: 20, ID-TH: 1, TW: 30
7. #Neighbors: 20, ID-TH: 2, TW: 15
8. #Neighbors: 20, ID-TH: 2, TW: 30

#Neighbors: 10, ID-TH:1 Time Window: 30

In this experiment, there are 33420 extracted patterns in total. Their confidence distribution (Figure 4.9) shows an important decrease in the number of patterns in the groups with a confidence value higher than 0,7. 11162 patterns contain the reference station in the head of the rule, but the majority of them does not contain the "target" state (they contain the "Increase" state). 7372 patterns contain the target state in the head of the rule. Among those, the rule with the highest confidence is

$$['Full\_T0\_0'],$$
$$['Decrease\_T1\_4, Decrease\_T1\_4, Decrease\_T1\_4'],$$
$$['AlmostFull\_T2\_0']$$

and has a confidence of 0.278 and a support of 413. These very low values will make the rules generated very difficult to apply for the next classification steps.

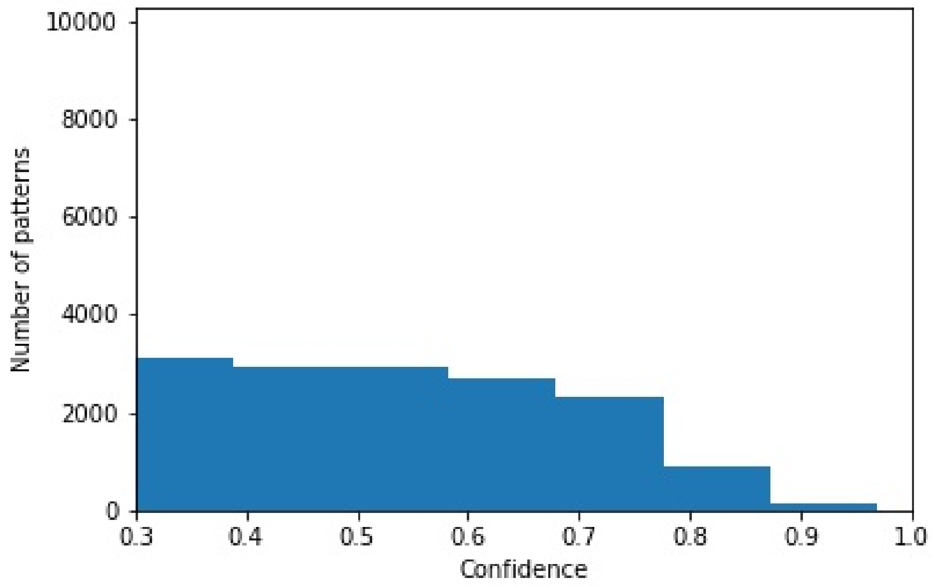

Figure 4.9: Confidence distribution of the patterns extracted in the experiment "Full-Decrease-wrapped" with 10 neighbors, ID-TH=1, and Time Window = 15.

The parameters used showed to have an impact on the patterns extracted.

Changing the number of likely departing stations

Also in this case, increasing this value allows extracting more patterns. We can compare the experiment above with the experiment #6. Doubling the number of stations considered, the number of patterns grows from 33420 to 48944. Also the number of patterns with the target state in the head of the rule increases to 10137, but the maximum confidence value of a rule is still very low (0.252). Some of the patterns excluded in the previous experiment showed interesting confidence ranges, however they do not contain the target state, as we can see in Figure 4.10.

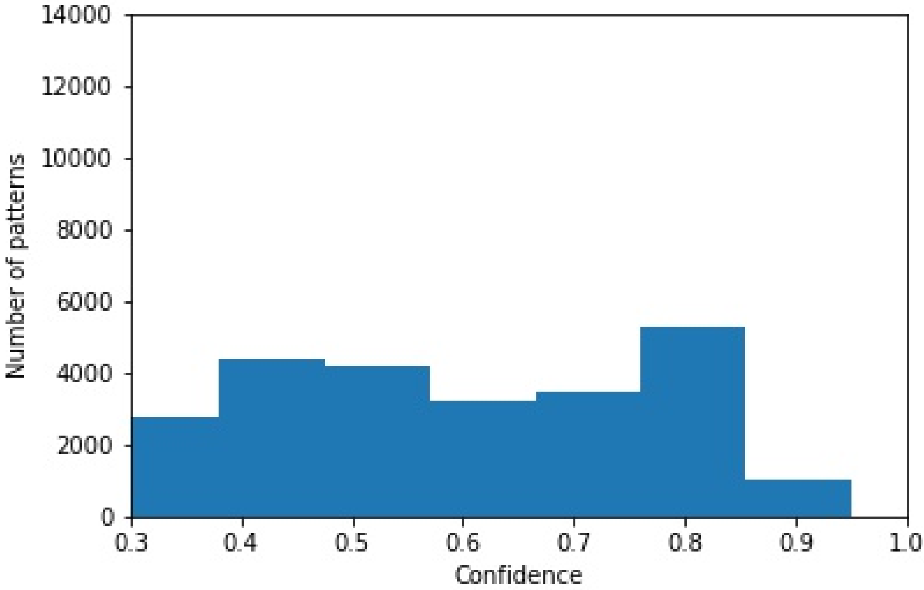

Figure 4.10: Confidence distribution of the patterns extracted in the experiment "Full-Decrease-wrapped" with 20 neighbors, ID-TH=1, and Time Window = 30.

Changing the ID-TH value

Increasing this parameter from 1 to 2 in all the cases leads to a reduction in the number of extracted patterns. As an example, we can compare the results of the experiment #2 with the experiment #4. In this case, the number of extracted patterns drops from 33420 to 23177, and also the patterns with the target state for the reference station are reduced. Comparing the patterns distribution, as we can see in Figure 4.11, there are far less patterns in the ranges with higher confidence values when the threshold value is 2. Considering that the threshold for the "AlmostFull" state is 3, probably 2 is not enough to spot the small differences in the stations that may have an impact on the reference station.

Changing the Time Window

In this set of experiments, changing this parameter shows an increase in the quantity of patterns extracted when passing from a time window of width 15 to a window of width 30. The total number of patterns goes from 13518 to 33420 in the experiments #1 and #2. If we compare the patterns distribution (Figure 4.12), more patterns are present when the time window is wider.

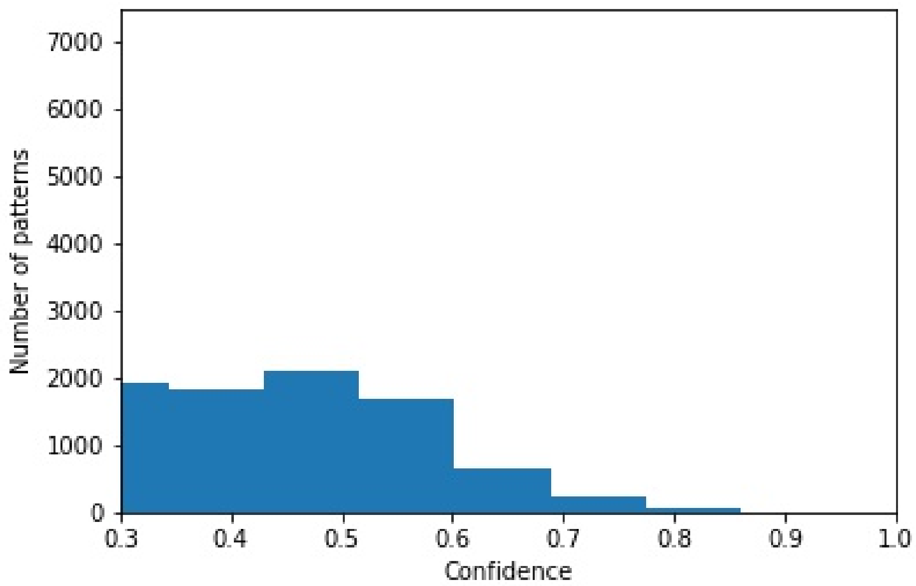

Figure 4.11: Confidence distribution of the patterns extracted in the experiment "Full-Decrease-wrapped" with 10 neighbors, ID-TH=2, and Time Window = 30.

4.1.4 Full-Decrease-wrapped

In these experiments, the events considered for the event generation are the same as the "Full-Decrease" experiment, but the events "AlmostFull" and "Full" are considered as the same event (see Wrap States parameter in Section 3.3.2). This means that all the cases where the number of docks available is below the specified threshold, the event is considered as "AlmostFull", even if the actual value is zero. The time window values considered are 15 and 30. Also the threshold for "increase" and "decrease" states values were changed.

For this experiment these fixed values were used for the parameters:

- Neighborhood type: "indegree";
- discrete distance unit: 500 meters;
- minimum support for PrefixSpan: 0.001;

Here all the tested configurations are listed:

1. #Neighbors: 20, ID-TH: 0, TW: 15
2. #Neighbors: 20, ID-TH: 0, TW: 30

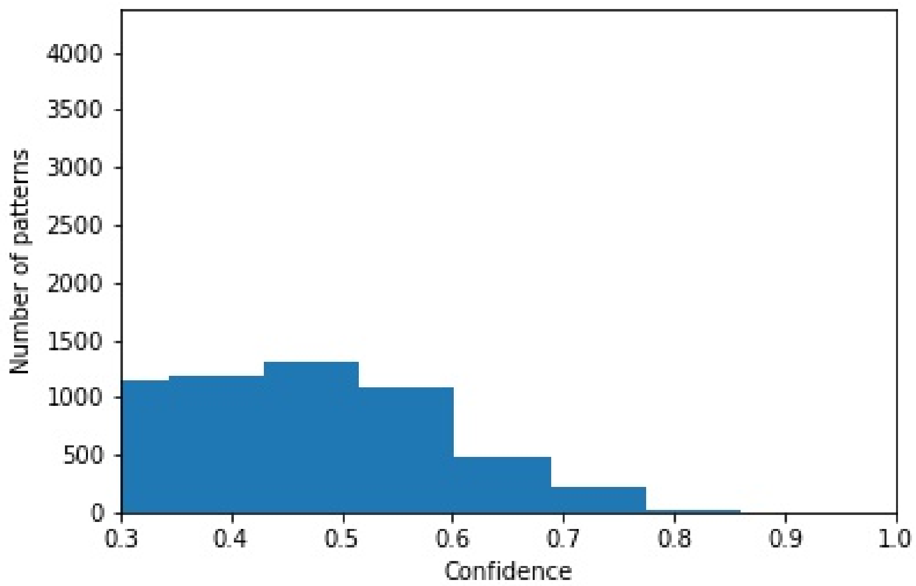

Figure 4.12: Confidence distribution of the patterns extracted in the experiment "Full-Decrease-wrapped" with 10 neighbors, ID-TH=1, and Time Window = 15.

3. #Neighbors: 20, ID-TH: 1, TW: 15
4. #Neighbors: 20, ID-TH: 1, TW: 30
5. #Neighbors: 20, ID-TH: 2, TW: 15
6. #Neighbors: 20, ID-TH: 2, TW: 30

Changing the ID-TH value

The results obtained by changing this value do not differ too much from the previous findings. Again, varying the number from 0 to 2 leads to a progressive decrease in the number of patterns, and especially their quality, as we can see in Figures 4.13, 4.14, and 4.15.

Changing the Time Window

Also in this case, the results are not different from the previous experiments, showing that the number of extracted rules grows when dealing with wider time windows. This growth can have quite big factors. For instance, in the experiments #5 and #6, we have an increase from 7163 to 24362 (more than 3 times).

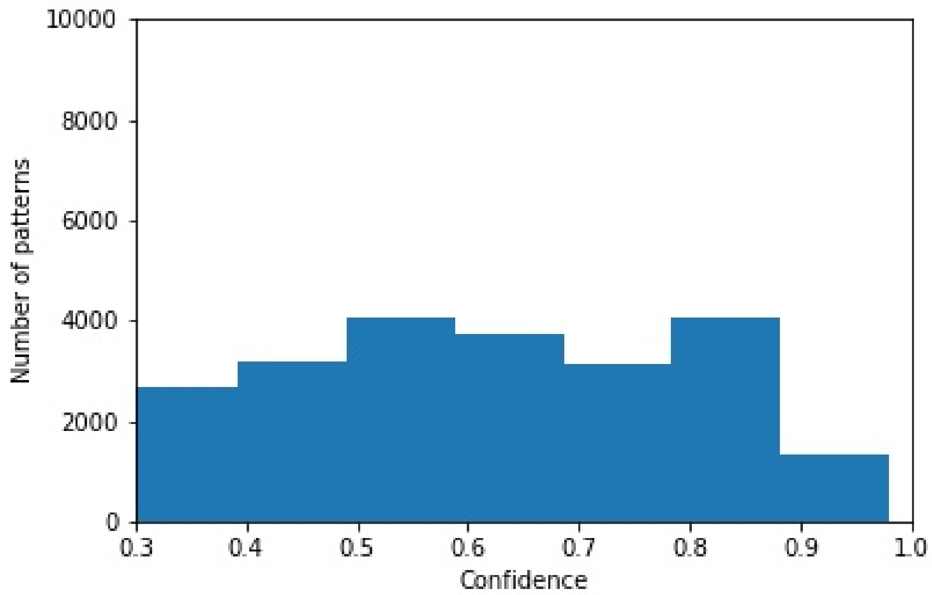

Figure 4.13: Confidence distribution of the patterns extracted in the experiment "Full-Decrease-wrapped" with 20 neighbors, ID-TH=0, and Time Window = 15.

4.1.5 Empty-AlmostEmpty

In these experiments, the events considered for the event generation are "Empty" and "AlmostEmpty" only. The target of this and the following event generation in this section is the "Empty" state. The experiments were run both for "distance" and "indegree" neighborhood definitions, and the time window values considered are 15 and 30 For this experiment these values were used for the parameters:

- minimum support for PrefixSpan: 0.05;
- discrete distance unit: 500 meters;
- maximum discrete distance in the "distance" case: 3;
- number of likely departing stations in the "indegree" case: 20;

Neighborhood: Distance, Time Window: 15

In this experiment, there are 1819 extracted patterns in total. Their confidence distribution (Figure 4.16) shows that the rules are mostly uniformly distributed with some peaks around 0,6 and 0,8. Only 265 patterns contain the reference

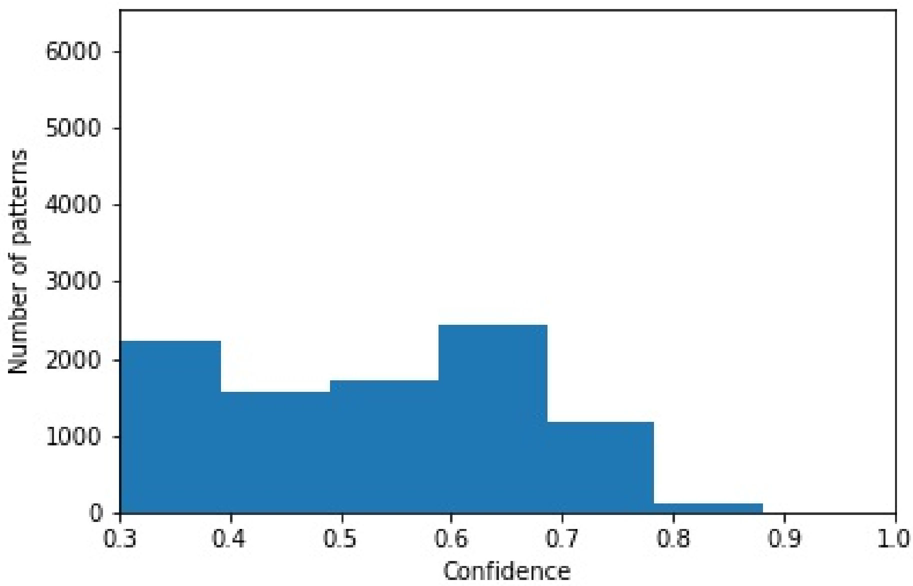

Figure 4.14: Confidence distribution of the patterns extracted in the experiment "Full-Decrease-wrapped" with 20 neighbors, ID-TH=1, and Time Window = 15.

station in the head of the rule. Among those, the rule with the highest confidence is

$$['AlmostEmpty\_T0\_0'],$$
$$['AlmostEmpty\_T1\_0'],$$
$$['AlmostEmpty\_T2\_0']$$

and has a confidence of 0.857 and a support of 180045.

Neighborhood: Distance, Time Window: 30

In this experiment, there are 4734 extracted patterns in total. Their confidence distribution (Figure 4.17) shows a decreasing number of patterns when the confidence increases, with the exceptions of the ranges 0.6 - 0.7 and 0.8 - 0.9. Only 552 patterns contain the reference station in the head of the rule. Among those, the

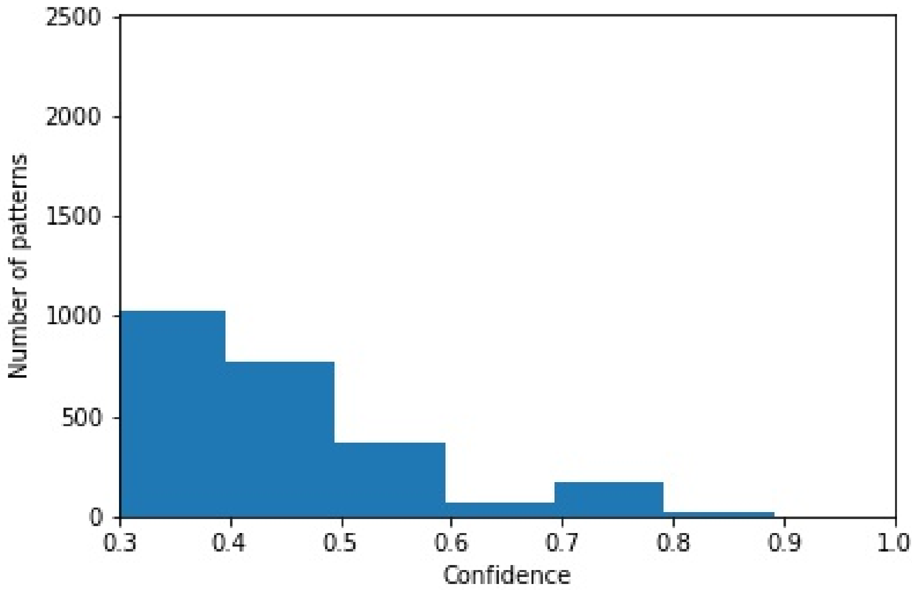

Figure 4.15: Confidence distribution of the patterns extracted in the experiment "Full-Decrease-wrapped" with 20 neighbors, ID-TH=2, and Time Window = 15.

rule with higher confidence is again

$$['AlmostEmpty\_T0\_0'],\\ ['AlmostEmpty\_T1\_0'],\\ ['AlmostEmpty\_T2\_0']$$

and has a confidence of 0.783 and a support of 92369.

Neighborhood: Indegree, Time Window: 15

In this experiment, there are 612 extracted patterns in total. Their confidence distribution (Figure 4.18) shows that the majority of the patterns are in the range 0.6 - 0.85. Only 129 patterns contain the reference station in the head of the rule. Among those, the rule with higher confidence is

$$['AlmostEmpty\_T0\_0, AlmostEmpty\_T0\_1'],\\ ['AlmostEmpty\_T1\_0, AlmostEmpty\_T1\_1'],\\ ['AlmostEmpty\_T2\_0']$$

and has a confidence of 0.861 and a support of 16338.

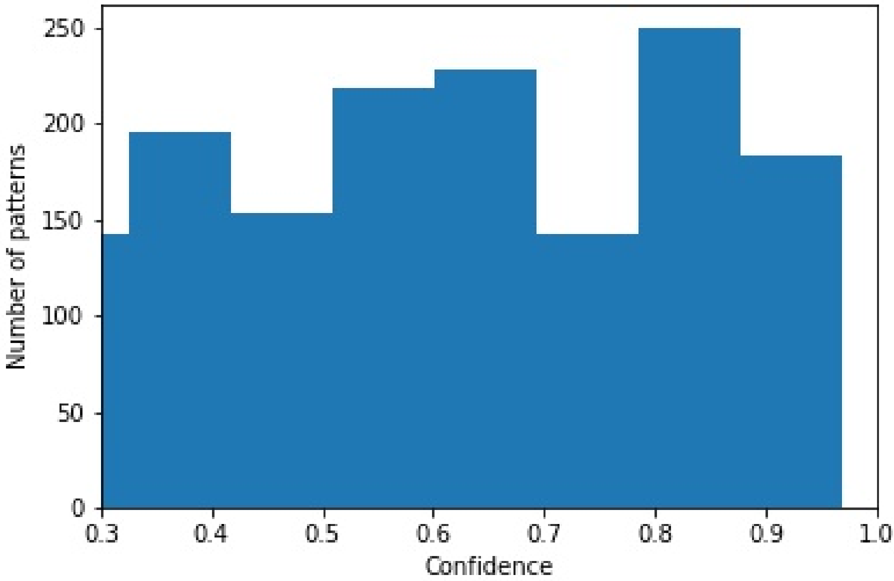

Figure 4.16: Confidence distribution of the patterns extracted in the experiment "Empty-AlmostEmpty" with Neighborhood Distance and Time Window = 15.

Neighborhood: Indegree, Time Window: 30

In this experiment, there are 1796 extracted patterns in total. Their confidence distribution (Figure 4.19) shows the higher group of patterns near to 0.8. Only 294 patterns contain the reference station in the head of the rule. Among those, the rule with higher confidence is once more

$$['AlmostEmpty\_T0\_0'],$$
$$['AlmostEmpty\_T1\_0'],$$
$$['AlmostEmpty\_T2\_0']$$

and has a confidence of 0.783 and a support of 92369, as in Section 4.1.5.

4.1.6 Empty-Increase

In these experiments, the events considered for the event generation are "Empty", "AlmostEmpty" and "Decrease" for the reference station, and "Increase" for the neighbors. The purpose of this kind of extraction is to highlight the impact of "Increase" events in the departing stations on the reference station. It corresponds to the opposite of the "Full-Decrease" case. The time window values considered are

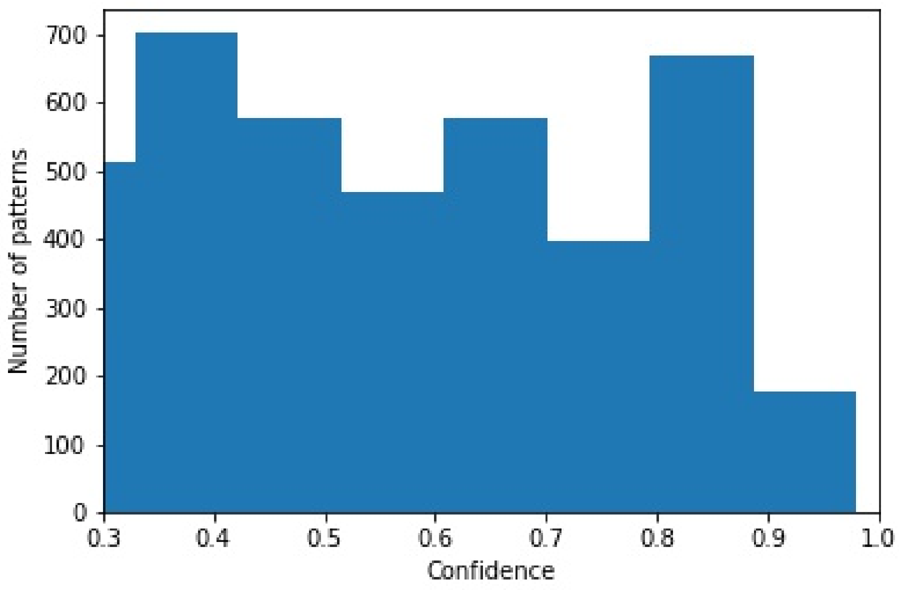

Figure 4.17: Confidence distribution of the patterns extracted in the experiment "Empty-AlmostEmpty" with Neighborhood Distance and Time Window = 30.

15 and 30. Also the threshold for "increase" and "decrease" states and the minimum support for PrefixSpan values were changed. Different values for the number of likely arriving stations were considered. For this experiment these fixed values were used for the parameters:

- Neighborhood type: "indegree;
- discrete distance unit: 500 meters;

Here all the tested configurations are listed:

1. #Neighbors: 10, ID-TH: 0, TW: 15, MS: 0.1
2. #Neighbors: 10, ID-TH: 1, TW: 15, MS: 0.1
3. #Neighbors: 10, ID-TH: 1, TW: 15, MS: 0.15
4. #Neighbors: 10, ID-TH: 2, TW: 15, MS: 0.15
5. #Neighbors: 20, ID-TH: 0, TW: 15, MS: 0.001
6. #Neighbors: 20, ID-TH: 0, TW: 15, MS: 0.1

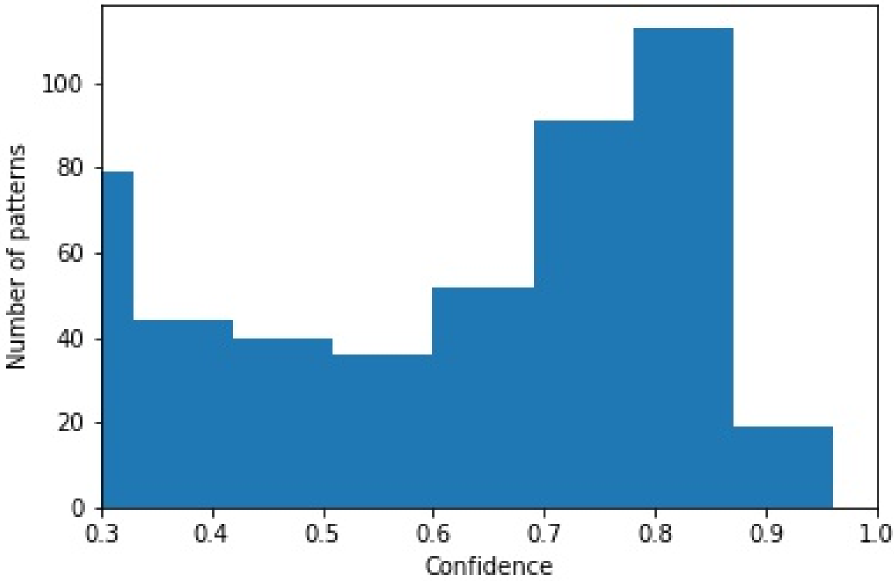

Figure 4.18: Confidence distribution of the patterns extracted in the experiment "Empty-AlmostEmpty" with Neighborhood Indegree and Time Window = 15.

7. #Neighbors: 20, ID-TH: 0, TW: 30, MS: 0.001
8. #Neighbors: 20, ID-TH: 1, TW: 15, MS: 0.001
9. #Neighbors: 20, ID-TH: 1, TW: 15, MS: 0.1
10. #Neighbors: 20, ID-TH: 1, TW: 15, MS: 0.15
11. #Neighbors: 20, ID-TH: 1, TW: 30, MS: 0.001
12. #Neighbors: 20, ID-TH: 2, TW: 15, MS: 0.001
13. #Neighbors: 20, ID-TH: 2, TW: 15, MS: 0.15
14. #Neighbors: 20, ID-TH: 2, TW: 30, MS: 0.001

The considered experiments have some aspects in common, and we can identify some general effects in changing the parameters.

Changing the number of likely departing stations

Changing the number of potentially departing station has obviously an impact on the number of the patterns extracted. Similarly to what we saw in the experiments

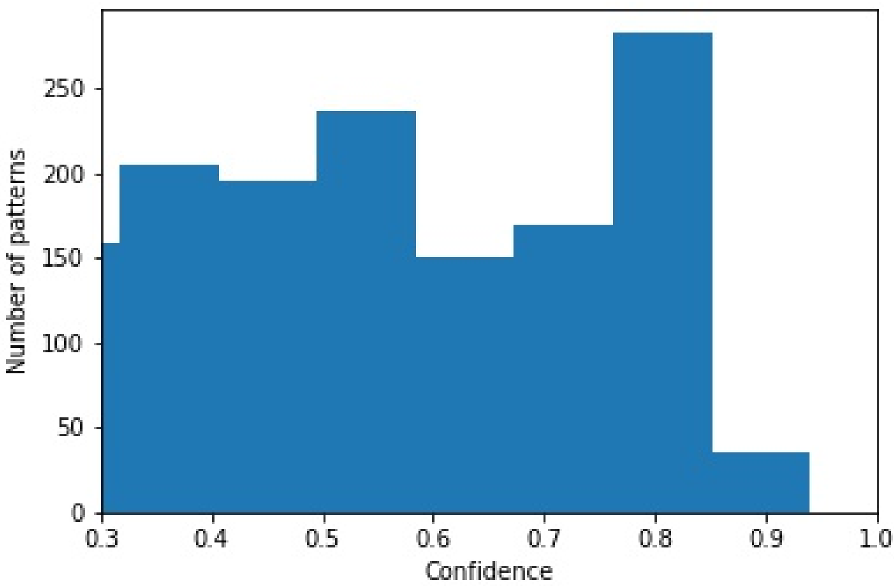

Figure 4.19: Confidence distribution of the patterns extracted in the experiment "Empty-AlmostEmpty" with Neighborhood Indegree and Time Window = 30

targeting the "Full" state, selecting a higher number of neighbors allows to extract more patterns. The reason is surely related to the fact that generating more events allows considering more patterns. We can compare for example the experiments #2 and #9. For these configurations, the only difference lies in the different number of likely departing stations (10 and 20). The number of extracted patterns in the former case is lower than the latter, having extracted respectively 45 and 77 patterns. However, the gap in the number of patterns containing the reference station in the head of the rule is lower (31 vs 40). The "missing" patterns have not the highest values of the confidence, but some still have some interesting values, as we can see in the Figures 4.20 and 4.21. Also the couples of experiments #1-#6 and #4-#13 share the same characteristics, with even higher reductions in the number of extracted patterns. In these cases, when the number of extracted patterns is higher, also the number of rules that contain the reference station in the head is decreased for the cases where less neighbors are considered. This reduction does not necessarily involve only lower-quality patterns, as we already noticed many times.

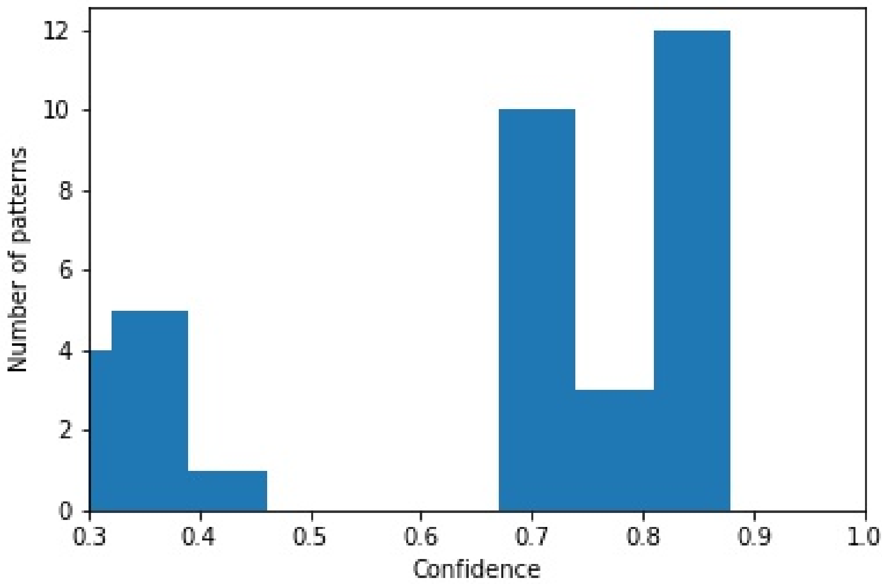

Figure 4.20: Confidence distribution of the patterns extracted in the experiment "Empty-Increase" #2.

Changing the Time Window

Also changing the time window value impacts the number of extracted patterns. As a consequence, also the confidence of some patterns seems to increase. The results are summarized in Table 4.2. It is difficult to infer why more patterns are extracted with larger windows. As already mentioned, a possible explanation is that the difference lies in the data itself.

Experiment number	Patterns	Patterns with reference station
5	95723	31090
7	130551	35527
8	69517	18782
11	89516	28341
12	26713	6559
14	74397	19938

Table 4.2: Comparison of the pattern extracted with different time window values for the Empty-Increase experiment.

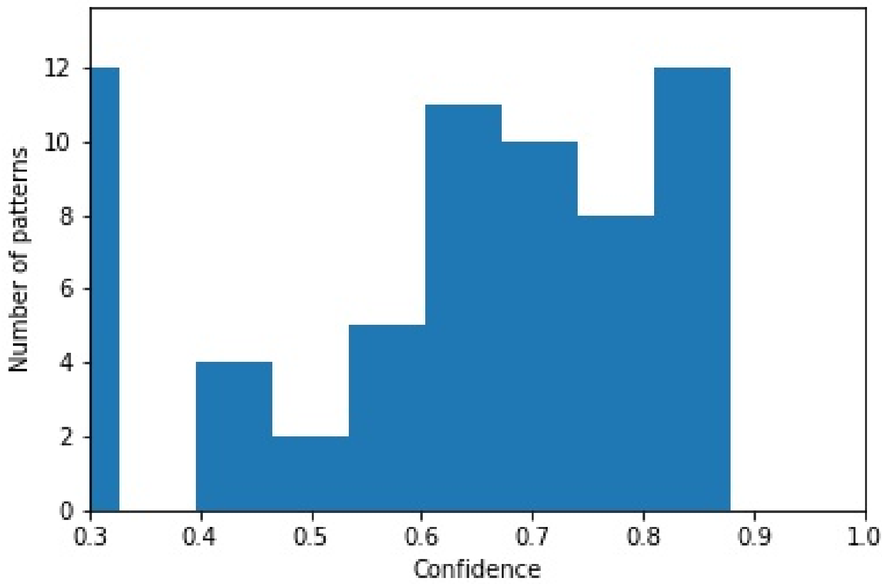

Figure 4.21: Confidence distribution of the patterns extracted in the experiment "Empty-Increase" #9.

Changing the ID-TH value

Changing this value, similarly to what we saw in Section 4.1.2, has a great impact on the number of patterns extracted, and consequently on their quality. Setting it equal to zero allows to extract the higher number of patterns. As we increase this value, the number of patterns extracted decreases. Since the threshold for the "Almost Critical" event was set to 3, values higher than 2 were not tested.
We can see an example of this in the experiments #5, #8, and #12. In these cases, increasing the parameter's value leads to a huge decrease in the number of patterns. The respective values in fact are: 95723, 69517, and 26713. If we consider only the rules with the reference station in the head, the situation is not different: in this case the values are 31090, 18782, and 6559.

4.1.7 Empty-Increase-stateChange

In these experiments, the events considered for the event generation are the same as the "Empty-Increase" experiment, but the definition of "Empty" and "AlmostEmpty" states is slightly changed (see State Change parameter in Section 3.3.2). The time window values considered are 15 and 30. Also the threshold for "increase" and "decrease" states values were changed. Different values for the number of likely arriving stations were considered. For this experiment these fixed values were used for the parameters:

- Neighborhood type: "indegree";
- discrete distance unit: 500 meters;
- minimum support for PrefixSpan: 0.001;

Here all the tested configurations are listed:

1. #Neighbors: 10, ID-TH: 1, TW: 15
2. #Neighbors: 10, ID-TH: 1, TW: 30
3. #Neighbors: 10, ID-TH: 2, TW: 15
4. #Neighbors: 10, ID-TH: 2, TW: 30
5. #Neighbors: 20, ID-TH: 1, TW: 15
6. #Neighbors: 20, ID-TH: 1, TW: 30
7. #Neighbors: 20, ID-TH: 2, TW: 15
8. #Neighbors: 20, ID-TH: 2, TW: 30

The results of these experiments, in general show a similar behaviour to their counterparts in the "Full" case. Also, the patterns extracted have similar effects on the extraction. To avoid repetitions, the specific results are not reported here, since they were already discussed in the previous sections. The pattern with the highest confidence having the reference station and the target state in the head of the rule is

$$\begin{aligned}['AlmostEmpty\_T0\_0, Empty\_T0\_0'],\\ ['Decrease\_T1\_0, Increase\_T1\_7'],\\ ['Empty\_T2\_0']\end{aligned}$$

and has a confidence of 0.396 and a support of 501. This means that using this kind of event generation will generate low quality rules for the classification.

4.1.8 Empty-Increase-wrapped

In these experiments, the events considered for the event generation are the same as the "Empty-Increase" experiment, but the events "AlmostEmpty" and "Empty" are considered as the same event, similarly to the "Full-Decrease-Wrapped" experiments. The time window values considered are 15 and 30. Also, the threshold for "increase" and "decrease" states values were changed. For this experiment these fixed values were used for the parameters:

- Neighborhood type: "indegree";
- discrete distance unit: 500 meters;
- minimum support for PrefixSpan: 0.001;
- number of likely arriving stations: 20;

Here all the tested configurations are listed:

1. #Neighbors: 20, ID-TH: 0, TW: 15
2. #Neighbors: 20, ID-TH: 0, TW: 30
3. #Neighbors: 20, ID-TH: 1, TW: 15
4. #Neighbors: 20, ID-TH: 1, TW: 30
5. #Neighbors: 20, ID-TH: 2, TW: 15
6. #Neighbors: 20, ID-TH: 2, TW: 30

The results of these experiments, in general show a similar behaviour to their counterparts in the "Full" case. Also the patterns extracted have similar effects on the extraction. To avoid repetitions, the specific results are not reported here, since they were already discussed in the previous sections.

4.1.9 Overall comments

There are some general differences and similarities in the various experiments performed. The most important difference is in the number of frequent patterns extracted. The experiments targeting the "Full" case constantly show a lower number of patterns than in those with target the "Empty" case, in all the cases seen. A common fact instead is the frequency of the "constant" cases. In fact, patterns with the form [AlmostCritical_T0_0], [AlmostCritical_T1_0], [AlmostCritical_T2_0] have high confidence and support values for both the critical target states. There are cases where the confidence has values higher than 0.8, meaning that in the dataset the critical events tend to remain constant in time, e.g. when a station has been in the "AlmostFull" state for 2 consecutive time periods, in most cases it will remain in the same state also in the next one.

4.2 Associative classification

In this section we present in detail all the experiments conducted with the associative classifier, all the different parameters considered, and the results. Before proceeding with the results, there are a few things that have to be clarified. First, the associative classifier is more resilient to missing data than the station-specific approach, that will be presented in the next section. In order to make all the results comparable, an additional pre-processing was performed. More specifically, at testing phase, all the patterns that were generated in a time period where at least one of all the stations did not contain any records were discarded. Second, in all the following tables, the column "Extraction Type" identifies a specific combination of parameters. To have a clearer representation, the meaning of these parameters is presented here:

- Full-AlmostFull: it is the experiment presented in Section 4.1.1;
- Full-Decrease: it is the experiment presented in Section 4.1.2;
- Full-Increase: in this kind of experiment, we generate only the events "Full", "AlmostFull" and "Increase" for all the stations;
- Empty-AlmostEmpty: it is the experiment presented in Section 4.1.5;
- Empty-Increase: it is the experiment presented in Section 4.1.6;
- Empty-Decrease: in this kind of experiment, we generate only the events "Empty", "AlmostEmpty" and "Decrease" for all the stations;
- _wrapped suffix: the "Critical" and the "AlmostCritical" states are considered as the same state. All the experiments done have this suffix;
- _neg suffix: when performing the event generation, generate also the "Not increasing" and "Not decreasing" events;
- Dummy: it consists in a classifier that predicts the next state as the current state. Basically, it always classifies the situation as stable. It constitutes the baseline of each experiment;

4.2.1 Results AlmostFull

Here, the experiments where the target is the "AlmostFull" state are presented. The class labels can contain two values, that represent the states "Normal" and "AlmostFull". We can group these experiments in some categories.

Experiments with different time intervals

In this case, the experiments were run with the following parameters setting:

- Support for PrefixSpan: 0.001;
- Spatial unit: 500 m;
- Window size: 3;
- AC-TH: 3;
- ID-TH: 1;
- #Neighbors when considering "indegree" distance (X): 10;
- Maximum distance in the other case of distance definition (Z): 3;
- State change: False;
- Wrap states: True;
- Time zone: whole day;

Time interval = 15
In this case all the tested configurations reach high values of precision, recall, and f1 score. However, they do not manage to overcome the value of the baseline, that has very high values of precision and recall (0.861). All the results are shown in Table 4.3. This is due to the fact that with these configurations, the rule constituting the baseline is extracted, and in many cases other rules extend the baseline rule by adding more contextual information related to neighbouring station. We can take as an example the rule with highest confidence in the Full-Increase event generation:

$$['AlmostFull\_T0\_0, AlmostFull\_T0\_3, AlmostFull\_T0\_2, AlmostFull\_T0\_4'], \\ ['AlmostFull\_T1\_0']$$

This rule has a 0.945 confidence, but a support of only 261, and is only a special case of the Baseline rule:

$$['AlmostFull\_T0\_0,], \\ ['AlmostFull\_T1\_0']$$

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	10	0.5	1	0.982224	0.815249	0.869150	0.841337
Full-AlmostFull_wrapped	10	0.5	5	0.961428	0.839290	0.357033	0.500958
Full-AlmostFull_wrapped	10	0.5	10	0.959953	0.853235	0.315795	0.460975
Full-AlmostFull_wrapped	10	0.8	1	0.983097	0.829122	0.866965	0.847621
Full-AlmostFull_wrapped	10	0.8	5	0.960860	0.851401	0.337008	0.482879
Full-AlmostFull_wrapped	10	0.8	10	0.949599	0.844535	0.086430	0.156811
Full-AlmostFull_wrapped	10	0.9	1	0.953792	0.880040	0.171188	0.286621
Full-AlmostFull_wrapped	Dist	0.5	1	0.981255	0.800810	0.870950	0.834409
Full-AlmostFull_wrapped	Dist	0.5	5	0.967110	0.823750	0.500546	0.622708
Full-AlmostFull_wrapped	Dist	0.5	10	0.965426	0.832851	0.453394	0.587151
Full-AlmostFull_wrapped	Dist	0.8	1	0.983754	0.840665	0.864200	0.852270
Full-AlmostFull_wrapped	Dist	0.8	5	0.966533	0.844578	0.469144	0.603215
Full-Decrease_wrapped	10	0.5	1	**0.985002**	**0.861674**	**0.861758**	**0.861716**
Full-Decrease_wrapped	10	0.5	5	0.959460	0.733191	0.396760	0.514891
Full-Decrease_wrapped	10	0.7	1	**0.985002**	**0.861674**	**0.861758**	**0.861716**
Full-Decrease_wrapped	10	0.7	5	0.961602	0.798973	0.390010	0.524158
Full-Decrease_wrapped	10	0.8	1	**0.985002**	**0.861674**	**0.861758**	**0.861716**
Full-Decrease_wrapped	10	0.8	5	0.949025	0.834110	0.074826	0.137333
Full-Decrease_wrapped	Dist	0.5	1	0.980669	0.792457	0.871850	0.830260
Full-Decrease_wrapped	Dist	0.5	5	0.960515	0.732655	0.428002	0.540345
Full-Decrease_wrapped	Dist	0.7	1	**0.985002**	**0.861674**	**0.861758**	**0.861716**
Full-Decrease_wrapped	Dist	0.7	5	0.962948	0.801002	0.421381	0.552244
Full-Decrease_wrapped	Dist	0.8	1	**0.985002**	**0.861674**	**0.861758**	**0.861716**
Full-Decrease_wrapped	Dist	0.8	5	0.949198	0.834925	0.078683	0.143814
Full-Increase_wrapped	10	0.5	1	0.980033	0.783375	0.873264	0.825881
Full-Increase_wrapped	10	0.5	5	0.967727	0.760496	0.590930	0.665075
Full-Increase_wrapped	10	0.7	1	0.982167	0.814363	0.869279	0.840925
Full-Increase_wrapped	10	0.7	5	0.970510	0.819726	0.584758	0.682587
Full-Increase_wrapped	10	0.8	1	0.982967	0.826954	0.867382	0.846686
Full-Increase_wrapped	10	0.8	5	0.964096	0.842678	0.415435	0.556512
Full-Increase_wrapped	0	0.5	1	0.984453	0.850962	0.864747	0.857799
Full-Increase_wrapped	0	0.5	5	0.950951	0.784047	0.131750	0.225592
Full-Increase_wrapped	0	0.7	1	**0.985002**	**0.861674**	**0.861758**	**0.861716**
Full-Increase_wrapped	0	0.7	5	0.950951	0.784047	0.131750	0.225592
Full-Increase_wrapped	0	0.8	1	**0.985002**	**0.861674**	**0.861758**	**0.861716**
Full-Increase_wrapped	0	0.8	5	0.946383	0.807760	0.014721	0.028915
Full-Increase_wrapped	Dist	0.5	1	0.979735	0.778843	0.874646	0.823969
Full-Increase_wrapped	Dist	0.5	5	0.970824	0.762339	0.671188	0.713866
Full-Increase_wrapped	Dist	0.7	1	0.981182	0.799675	0.871207	0.833910
Full-Increase_wrapped	Dist	0.7	5	0.973299	0.807932	0.665885	0.730063
Full-Increase_wrapped	Dist	0.8	1	0.983282	0.832818	0.865422	0.848807
Full-Increase_wrapped	Dist	0.8	5	0.969649	0.840374	0.543520	0.660109
Dummy	0	0.0	1	**0.985002**	**0.861674**	**0.861758**	**0.861716**

Table 4.3: Results of the experiments with Time interval = 15 with target state = "AlmostFull".

Time interval = 30

This is the configuration in which more tests were conducted. The results, presented in Table 4.4, show that the better results, both in terms of precision and f1 score, are achieved with the extractions that consider also the negative states. This means that, in addition to the information regarding the increase and decrease of the number of bicycles in a station, also the complementary information is

relevant to achieve the best results. The Dummy experiment, that constitutes the baseline of this configuration, is very difficult to be beaten. The best results of the configurations that do not consider the "negative" states are only able to reach its performance. Only one experiment, considering a confidence threshold equal to 0.8 and a number of matches equal to 5, is able to increase the precision. However, that little increase has a very high cost in terms of recall, leading to a huge drop in terms of recall. This happens because the confidence of the rules different from the "Dummy rule" is slightly higher, whereas their support is far lower. This results in a drop of the number of matches. The confidence threshold parameter is crucial to obtain satisfactory results: if we choose too low values indeed the recall will have high values, but negatively affecting the precision. Since the objective of this classification is to identify critical and rare events, a classifier that performs a positive prediction too frequently is less valuable than one that makes less positive predictions, but is more reliable about the quality of the prediction.
Time interval = 60
In this experiment, whose results are shown in table 4.5, the baseline has a lower value. However, no extracted rule seems to show different behaviour from what has been already discussed in the previous paragraphs.
In general, changing the size of the time window seems to have an impact on the difficulty of the problem, but the extracted rules seem to be not enough to achieve good results. The only event generations able to generate satisfactory results are those that consider the "negative" states, that demonstrated to hold useful information for the classification algorithm.

Experiments with different values for AlmostFull and time intervals

The goal of this kind of experiment is to investigate if, by changing the parameters related to the definition of the almost critical event, we are able to extract better quality rules and outperform the baseline. The tested configurations are:

- AC-TH = 1, TW = 15;
- AC-TH = 1, TW = 30;
- AC-TH = 2, TW = 15;
- AC-TH = 2, TW = 30;
- AC-TH = 3, TW = 15;
- AC-TH = 3, TW = 30;

The last two experiments were already presented in the previous paragraphs. In general, changing these parameters is effective for changing the baseline, however

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	10	0.50	1	0.968959	0.723582	0.805563	0.762375
Full-AlmostFull_wrapped	10	0.50	5	0.951936	0.739315	0.343565	0.469124
Full-AlmostFull_wrapped	10	0.50	10	0.950635	0.763784	0.291542	0.422003
Full-AlmostFull_wrapped	10	0.70	1	0.971048	0.747817	0.802122	0.774018
Full-AlmostFull_wrapped	10	0.70	5	0.951189	0.769131	0.300570	0.432228
Full-AlmostFull_wrapped	Dist	0.50	1	0.967362	0.706207	0.808215	0.753776
Full-AlmostFull_wrapped	Dist	0.50	5	0.956480	0.725902	0.475484	0.574594
Full-AlmostFull_wrapped	Dist	0.50	10	0.954311	0.735411	0.407437	0.524363
Full-AlmostFull_wrapped	Dist	0.70	1	0.972252	0.763162	0.799075	0.780706
Full-AlmostFull_wrapped	Dist	0.70	5	0.955479	0.755936	0.413135	0.534277
Full-AlmostFull_wrapped	Dist	0.80	1	0.974536	0.793875	0.794279	0.794077
Dummy	0	0.00	1	0.974536	0.793875	0.794279	0.794077
Full-Decrease_wrapped	10	0.50	1	0.965887	0.690565	0.811939	0.746350
Full-Decrease_wrapped	10	0.50	5	0.952550	0.649419	0.504937	0.568136
Full-Decrease_wrapped	10	0.70	1	0.974536	0.793875	0.794279	0.794077
Full-Decrease_wrapped	10	0.70	5	0.949735	0.706036	0.320093	0.440485
Full-Decrease_wrapped	10	0.80	1	0.974536	0.793875	0.794279	0.794077
Full-Decrease_wrapped	Dist	0.50	1	0.965423	0.686186	0.811939	0.743785
Full-Decrease_wrapped	Dist	0.50	5	0.951475	0.628543	0.525588	0.572473
Full-Decrease_wrapped	Dist	0.70	1	0.974536	0.793875	0.794279	0.794077
Full-Decrease_wrapped	Dist	0.70	5	0.953414	0.711777	0.413982	0.523492
Full-Decrease_wrapped	Dist	0.80	1	0.974536	0.793875	0.794279	0.794077
Full-Increase_wrapped	10	0.50	1	0.965224	0.684116	0.812616	0.742850
Full-Increase_wrapped	10	0.50	5	0.957460	0.662740	0.634881	0.648512
Full-Increase_wrapped	10	0.70	1	0.970968	0.746835	0.802291	0.773571
Full-Increase_wrapped	10	0.70	5	0.959574	0.743063	0.528861	0.617925
Full-Increase_wrapped	10	0.80	1	0.974498	0.793383	0.794279	0.793831
Full-Increase_wrapped	10	0.80	5	0.949466	0.795721	0.245500	0.375232
Full-Increase_wrapped	0	0.50	1	0.972541	0.765642	0.800937	0.782892
Full-Increase_wrapped	0	0.50	5	0.946645	0.708012	0.232861	0.350459
Full-Increase_wrapped	0	0.70	1	0.974536	0.793875	0.794279	0.794077
Full-Increase_wrapped	0	0.70	5	0.939750	0.689189	0.046042	0.086317
Full-Increase_wrapped	0	0.80	1	0.974536	0.793875	0.794279	0.794077
Full-Increase_wrapped_neg	10	0.50	1	0.964331	0.675238	**0.814873**	0.738513
Full-Increase_wrapped_neg	10	0.50	5	0.964331	0.675238	**0.814873**	0.738513
Full-Increase_wrapped_neg	10	0.60	1	0.964331	0.675238	**0.814873**	0.738513
Full-Increase_wrapped_neg	10	0.95	1	**0.978693**	0.943825	0.696778	**0.801701**
Full-Increase_wrapped_neg	10	0.90	1	0.977815	0.916008	0.705806	0.797285
Full-Decrease_wrapped_neg	Dist	0.50	1	0.964331	0.675238	**0.814873**	0.738513
Full-Decrease_wrapped_neg	Dist	0.70	1	0.965538	0.687071	0.812560	0.744565
Full-Decrease_wrapped_neg	Dist	0.90	1	0.977815	0.916008	0.705806	0.797285
Full-Decrease_wrapped_neg	Dist	0.95	1	0.977065	**0.949295**	0.664447	0.781731
Full-Decrease_wrapped_neg	10	0.50	1	0.964331	0.675238	**0.814873**	0.738513
Full-Decrease_wrapped_neg	10	0.70	1	0.964443	0.676300	0.814704	0.739078
Full-Decrease_wrapped_neg	10	0.90	1	0.977815	0.916008	0.705806	0.797285
Full-Decrease_wrapped_neg	10	0.95	1	0.978620	0.944006	0.695368	0.800832
Full-Increase_wrapped_neg	Dist	0.50	1	0.964331	0.675238	**0.814873**	0.738513
Full-Increase_wrapped_neg	Dist	0.70	1	0.965475	0.686392	0.812842	0.744285
Full-Increase_wrapped_neg	Dist	0.90	1	0.977815	0.916008	0.705806	0.797285
Full-Increase_wrapped_neg	Dist	0.95	1	0.977403	0.948111	0.671162	0.785953

Table 4.4: Results of the experiments with Time interval = 30 with target state = "AlmostFull"

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	10	0.5	5	0.935130	0.617621	0.328060	0.4285099317385155
Full-AlmostFull_wrapped	10	0.7	1	0.955736	**0.701100**	0.702354	0.7017263276729855
Full-AlmostFull_wrapped	10	0.7	5	0.933531	0.679178	0.195951	0.30415083308973984
Full-AlmostFull_wrapped	10	0.8	1	0.925915	0.705882	0.001130	0.0022562752655824014
Full-AlmostFull_wrapped	10	0.8	5	0.925859	0.000000	0.000000	Nan
Full-AlmostFull_wrapped	Dist	0.5	1	0.942683	0.593206	**0.721846**	0.65123391241558
Full-AlmostFull_wrapped	Dist	0.5	5	0.936617	0.600601	0.432863	0.5031191857283572
Full-AlmostFull_wrapped	Dist	0.7	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**
Full-AlmostFull_wrapped	Dist	0.7	5	0.937343	0.684388	0.287288	0.40469558296856356
Full-Decrease_wrapped	10	0.5	1	0.955604	0.699438	0.703390	0.7014084507042253
Full-Decrease_wrapped	10	0.5	5	0.943527	0.646412	0.525895	0.5799584631360333
Full-Decrease_wrapped	10	0.7	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**
Full-Decrease_wrapped	Dist	0.5	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**
Full-Decrease_wrapped	Dist	0.5	5	0.944525	0.651823	0.540301	0.5908459043402153
Full-Decrease_wrapped	Dist	0.7	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**
Full-Increase_wrapped	10	0.5	1	0.945265	0.611401	0.718079	0.6604598796172
Full-Increase_wrapped	10	0.5	5	0.946117	0.644775	0.608286	0.6259993216725616
Full-Increase_wrapped	10	0.7	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**
Full-Increase_wrapped	10	0.7	5	0.935060	0.682359	0.232015	0.3462862764387604
Full-Increase_wrapped	0	0.5	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**
Full-Increase_wrapped	0	0.5	5	0.935325	0.623835	0.321375	0.4242122925859176
Full-Increase_wrapped	0	0.7	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**
Full-Increase_wrapped	Dist	0.5	1	0.942683	0.593206	**0.721846**	0.65123391241558
Full-Increase_wrapped	Dist	0.5	5	0.944986	0.626911	0.637006	0.6319181729017795
Full-Increase_wrapped	Dist	0.7	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**
Full-Increase_wrapped	Dist	0.7	5	0.938501	0.685072	0.315443	0.4319793681495809
Dummy	0	0.0	1	**0.955743**	0.701241	0.702166	**0.7017032088077538**

Table 4.5: Results of the experiments with Time interval = 60 with target state = "AlmostFull".

the quality of the extracted rules, hence of the classification, varies accordingly to the baseline. As an example, the best performing experiments of two different configurations are presented in Tables 4.6 and 4.7. As we can see, none of the tested configuration is able to reach better results than the baseline in terms of f1 score. Some configurations, when dealing with confidence values higher than the "Dummy rule" or number of matches higher than 1 are able to reach slightly higher values of precision. The highest improvement, in Table 4.7, is of 5%. Nevertheless, the negative impact of this improvement is a very high decrease in the recall, meaning that more precise models are not able to catch the great majority of the positive cases. Such drops in the recall (from 73% to 11% in the best case), do not seem to be easily applicable in practical scenarios. Last, all kinds of event generation tested are able to reach the baseline.

Experiments with different time zones

As a last investigation, the configuration tested in the experiments whose results are shown in Table 4.4 were tested in different time zones. The 24 hours of the day were divided in 6 different groups of equal duration, to verify if whether there are significant variations, since the results of the data exploration showed a non-uniform

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	Dist	0.7	5	0.988976	0.723757	0.020450	0.039775
Full-Decrease_wrapped	10	0.5	1	**0.993813**	0.722846	**0.723072**	**0.722959**
Full-Decrease_wrapped	10	0.7	1	**0.993813**	0.722846	**0.723072**	**0.722959**
Full-Decrease_wrapped	Dist	0.5	1	**0.993813**	0.722846	**0.723072**	**0.722959**
Full-Decrease_wrapped	Dist	0.7	1	**0.993813**	0.722846	**0.723072**	**0.722959**
Full-Decrease_wrapped	Dist	0.7	5	0.989248	**0.729207**	0.058851	0.108912
Full-Increase_wrapped	10	0.7	1	**0.993813**	0.722846	**0.723072**	**0.722959**
Full-Increase_wrapped	0	0.5	1	**0.993813**	0.722846	**0.723072**	**0.722959**
Full-Increase_wrapped	0	0.7	1	**0.993813**	0.722846	**0.723072**	**0.722959**
Full-Increase_wrapped	Dist	0.7	1	**0.993813**	0.722846	**0.723072**	**0.722959**
Dummy	0	0.0	1	**0.993813**	0.722846	**0.723072**	**0.722959**

Table 4.6: Results of the experiments with Time interval = 15 and threshold for "AlmostFull" = 1.

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	10	0.7	5	0.966922	0.735984	0.110600	0.192301
Full-AlmostFull_wrapped	10	0.8	1	0.964945	0.750799	0.023021	0.044673
Full-AlmostFull_wrapped	10	0.8	5	0.964495	0.780000	0.003821	0.007604
Full-AlmostFull_wrapped	Dist	0.7	1	**0.981062**	0.733895	**0.734326**	**0.734110**
Full-Decrease_wrapped	10	0.5	1	**0.981062**	0.733895	**0.734326**	**0.734110**
Full-Decrease_wrapped	10	0.7	1	**0.981062**	0.733895	**0.734326**	**0.734110**
Full-Decrease_wrapped	Dist	0.5	1	**0.981062**	0.733895	**0.734326**	**0.734110**
Full-Decrease_wrapped	Dist	0.7	1	**0.981062**	0.733895	**0.734326**	**0.734110**
Full-Increase_wrapped	10	0.8	1	0.966462	0.765233	0.083660	0.150830
Full-Increase_wrapped	10	0.8	5	0.964533	**0.882353**	0.004408	0.008773
Full-Increase_wrapped	0	0.5	1	**0.981062**	0.733895	**0.734326**	**0.734110**
Full-Increase_wrapped	0	0.7	1	**0.981062**	0.733895	**0.734326**	**0.734110**
Full-Increase_wrapped	Dist	0.7	1	**0.981062**	0.733895	**0.734326**	**0.734110**
Dummy	0	0.0	1	**0.981062**	0.733895	**0.734326**	**0.734110**

Table 4.7: Results of the experiments with Time interval = 30 and threshold for "AlmostFull" = 2.

usage of the bike sharing system. These experiments are focused on assessing how the classifier performs in those situations, and how the scenario varies.

Hours 0 - 4

In this time zone the situation is nearly always constant. In fact, the total number of situations where the situation changes from AlmostFull to Normal or vice versa is 86 over more than 35000 samples. In this situation there is not much practical interest in predicting the critical situations because of the very low usage of the system, however it may be an interesting extreme situation where to test the classifier. A selection of the best results is presented in Table 4.8. As we expected, the precision and recall scores for the baseline are very close to 1. It is interesting to notice that we are still able to find stricter rules that allow more precise results, but again these rules are applied very rarely.

Hours 4 - 8

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	10	0.7	5	0.966193	0.980527	0.416951	0.585099
Full-AlmostFull_wrapped	10	0.8	1	0.944639	**1.000000**	0.031661	0.061379
Full-AlmostFull_wrapped	10	0.8	5	0.942996	**1.000000**	0.002923	0.005828
Full-AlmostFull_wrapped	Dist	0.5	5	0.974882	0.982397	0.570872	0.722120
Full-AlmostFull_wrapped	Dist	0.7	5	0.974798	0.982353	0.569411	0.720937
Full-AlmostFull_wrapped	Dist	0.8	1	0.949680	0.995968	0.120312	0.214689
Full-Decrease_wrapped	10	0.5	5	0.947118	0.981250	0.076473	0.141889
Full-Decrease_wrapped	10	0.7	1	**0.997605**	0.980459	**0.977594**	**0.979024**
Full-Decrease_wrapped	10	0.8	1	0.943136	**1.000000**	0.005358	0.010659
Full-Decrease_wrapped	Dist	0.7	1	**0.997605**	0.980459	**0.977594**	**0.979024**
Full-Decrease_wrapped	Dist	0.8	1	0.942913	**1.000000**	0.001461	0.002918
Full-Increase_wrapped	10	0.7	5	0.967140	0.981257	0.433512	0.601351
Full-Increase_wrapped	10	0.8	1	0.945865	0.990991	0.053580	0.101664
Full-Increase_wrapped	0	0.5	1	**0.997605**	0.980459	**0.977594**	**0.979024**
Full-Increase_wrapped	0	0.7	1	**0.997605**	0.980459	**0.977594**	**0.979024**
Full-Increase_wrapped	Dist	0.7	5	0.975745	0.982843	0.585972	0.734208
Full-Increase_wrapped	Dist	0.8	1	0.950042	0.985019	0.128105	0.226724
Full-Increase_wrapped	Dist	0.8	5	0.943163	**1.000000**	0.005845	0.011622
Dummy	0	0.0	1	**0.997605**	0.980459	**0.977594**	**0.979024**

Table 4.8: Results of the experiments in the time zone 0 - 4.

In this situation, the states are far more dynamic than in the previous time zone. Unfortunately, the rules extracted cannot adapt to the situation better than the baseline, and in the sole situation where we get a slightly higher precision score, the recall decreases to 11%. The best results are presented in Table 4.9.

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	Dist	0.5	1	**0.945100**	0.6653076352853966	**0.629825**	**0.6470800288392213**
Full-AlmostFull_wrapped	Dist	0.5	5	0.925165	**0.6666666666666666**	0.127018	0.2133804892425582
Full-Decrease_wrapped	10	0.5	1	**0.945100**	0.6653076352853966	**0.629825**	**0.6470800288392213**
Full-Decrease_wrapped	Dist	0.5	1	**0.945100**	0.6653076352853966	**0.629825**	**0.6470800288392213**
Full-Increase_wrapped	0	0.5	1	**0.945100**	0.6653076352853966	**0.629825**	**0.6470800288392213**
Full-Increase_wrapped	Dist	0.5	1	**0.945100**	0.6653076352853966	**0.629825**	**0.6470800288392213**
Dummy	0	0.0	1	**0.945100**	0.6653076352853966	**0.629825**	**0.6470800288392213**

Table 4.9: Results of the experiments in the time zone 4 - 8.

Hours 8 - 12
In this case the situation is less dynamic than in the previous time zone. As usual, the classifiers do not achieve better results compared to the baseline in terms of f1 score. Nevertheless, if we craft the confidence threshold precisely to exclude the baseline rule, we are able to reach a good enhancement in the precision (7%). The recall still assumes low values, but the reduction is the lowest obtained. This is a strategy that we can use if we want to augment the precision and limit the loss in the recall to some extent. The results are shown in Table 4.10

Hours 12 - 16
In this case the situation is not much different from the previous one. The results (Table 4.11) confirm that the exclusion of the baseline rule is an effective method

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	10	0.500	5	0.953272	0.757322	0.188150	0.301415
Full-AlmostFull_wrapped	Dist	0.500	1	**0.971679**	0.726185	**0.756757**	**0.741156**
Full-AlmostFull_wrapped	Dist	0.500	5	0.959955	0.728814	0.402287	0.518419
Full-Decrease_wrapped	10	0.500	1	**0.971679**	0.726185	**0.756757**	**0.741156**
Full-Decrease_wrapped	10	0.500	5	0.959844	0.735352	0.391372	0.510855
Full-Decrease_wrapped	Dist	0.500	1	**0.971679**	0.726185	**0.756757**	**0.741156**
Full-Decrease_wrapped	Dist	0.500	5	0.962852	0.735247	0.479210	0.580239
Full-Increase_wrapped	10	0.500	1	**0.971679**	0.726185	**0.756757**	**0.741156**
Full-Increase_wrapped	10	0.500	5	0.964884	0.730049	0.546778	0.625260
Full-Increase_wrapped	0	0.500	1	**0.971679**	0.726185	**0.756757**	**0.741156**
Full-Increase_wrapped	0	0.500	5	0.947452	0.793651	0.025988	0.050327
Full-Increase_wrapped	Dist	0.500	5	0.968087	0.729634	0.642412	0.683250
Dummy	0	0.000	1	**0.971679**	0.726185	**0.756757**	**0.741156**
Full-Decrease_wrapped	10	0.575	1	0.955194	**0.796610**	0.219854	0.344603

Table 4.10: Results of the experiments in the time zone 8 - 12.

to improve the precision.

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	Dist	0.5	1	0.962644	0.691805	**0.657116**	0.674014
Full-AlmostFull_wrapped	10	0.5000	5	0.945159	0.717391	0.110315	0.191225
Full-Decrease_wrapped	10	0.5000	1	**0.963149**	0.698324	0.656638	**0.676840**
Full-Decrease_wrapped	Dist	0.5000	1	**0.963149**	0.698324	0.656638	**0.676840**
Full-Increase_wrapped	10	0.5	1	0.963065	0.697062	**0.657116**	0.676500
Full-Increase_wrapped	Dist	0.5	1	0.962644	0.691805	**0.657116**	0.674014
Full-Increase_wrapped	0	0.5000	1	**0.963149**	0.698324	0.656638	**0.676840**
Full-Increase_wrapped	0	0.5000	5	0.942941	**0.790476**	0.039637	0.075489
Dummy	0	0.0000	1	**0.963149**	0.698324	0.656638	**0.676840**
Full-Decrease_wrapped	10	0.5931	1	0.952428	0.764940	0.275072	0.404636

Table 4.11: Results of the experiments in the time zone 12 - 16.

Hours 16 - 20
The results of the experiments (Table 4.12) in this time zone do not show big differences from the previous ones. All the kinds of the event generation are able to reach the same scores as the baseline. In 3 cases the precision score is improved by more than 10%, but the number of positive predictions decreases to very low values. Similarly to what happened in the previous two time zones, crafting the confidence threshold in order to exclude the baseline rule allows to increase the precision without nullifying the recall.
Hours 20 - 24
In this last time zone we have similar results to the other night time zone. In this case again the system has low numbers of user, and the situation tends to remain constant.

To conclude the analysis by time zone, we can say that for sure there are some time zones that need to be monitored with more attention, and they coincide with

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-Decrease_wrapped	10	0.70	1	0.939855	**0.846154**	0.010082	0.019928
Full-Decrease_wrapped	Dist	0.50	1	**0.968677**	0.717705	0.796975	**0.755266**
Full-Increase_wrapped	10	0.50	1	0.963202	0.661157	**0.806599**	0.726672
Full-Increase_wrapped	10	0.70	1	0.946359	0.776316	0.162236	0.268385
Full-Increase_wrapped	10	0.70	5	0.940022	0.815789	0.014207	0.027928
Full-Increase_wrapped	10	0.80	1	0.939689	0.750000	0.008249	0.016319
Full-Increase_wrapped	0	0.50	1	**0.968677**	0.717705	0.796975	**0.755266**
Full-Increase_wrapped	Dist	0.70	1	0.942023	0.811688	0.057287	0.107021
Dummy	0	0.00	1	**0.968677**	0.717705	0.796975	**0.755266**
Full-Decrease_wrapped	10	0.65	1	0.953113	0.761352	0.330431	0.460850

Table 4.12: Results of the experiments in the time zone 16 - 20.

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Full-AlmostFull_wrapped	10	0.70	5	0.964927	0.9707750952986023	0.382382	0.5486535008976661
Full-AlmostFull_wrapped	10	0.80	1	0.944475	**1.0**	0.004004	0.007976071784646063
Full-Decrease_wrapped	10	0.70	5	0.945954	**1.0**	0.030531	0.05925206410879067
Full-AlmostFull_wrapped	Dist	0.5	1	0.994503	0.9446913580246914	**0.957457**	0.9510315684812329
Full-Decrease_wrapped	Dist	0.70	1	**0.995675**	0.9675291730086251	0.954454	**0.9609473418997228**
Full-Decrease_wrapped	Dist	0.70	5	0.946010	0.9701492537313433	0.032533	0.06295399515738499
Full-Increase_wrapped	Dist	0.5	1	0.994392	0.9428289797930015	**0.957457**	0.9500869133349888
Full-Increase_wrapped	0	0.70	1	**0.995675**	0.9675291730086251	0.954454	**0.9609473418997228**
Dummy	0	0.00	1	**0.995675**	0.9675291730086251	0.954454	**0.9609473418997228**
Full-Decrease_wrapped	10	0.75	1	0.945564	0.9795918367346939	0.024024	0.046897899364924285

Table 4.13: Results of the experiments in the time zone 20 - 24.

the hours in the daytime. The zone where the situation is more unstable is the early morning (4-8). However, the quality of the prediction of the associative model does not outperform the baseline in terms of f1 score, adapting to its value in all the situations. Higher values of precision can be reached, but lowering the number of times when the model performs a positive prediction.
Also the analysis of one-hour-long time zones and intervals of 15 minutes was performed, but since the results do not differ in the substance from the ones already presented, they are not discussed in more details.

4.2.2 Results AlmostEmpty

Here, the experiments where the target is the "AlmostEmpty" state are presented. The class labels can contain two values that represent the states "Normal" and "AlmostEmpty". The majority of the experiments done in this work regard the "AlmostFull" case. With the aim of comparing the performances, the most representative experiments were also run with the "AlmostEmpty" target. The starting parameters configuration were the same as the one presented in Section 4.2.1.

Experiments with different time intervals

Time interval = 30
In this experiment none of the configurations tested resulted in higher f1 scores compared to the baseline. 6 configurations, including all the types of event generation and distance definition, reach better results when considering the precision metric. All the tested configurations are presented in Table 4.14. As we can see, the baseline is not very different from the same experiment with opposite target. This means that the two situations tend to having similar behaviour in terms of variations between the normal and critical states.
Time interval = 15
The best performing results of the experiments run with this configuration are presented in Table 4.15. As we can see, no experiment reaches higher results than the baseline in terms of f1 score, and all the enhancements in precision happen only with high costs in recall, hence rules with higher precision loose their applicability in the most common cases.

Experiments with different values for AlmostEmpty

The thresholds tested are 1, 2 and 3, as we did in the similar section when we were discussing the "AlmostFull" case. We can analyze the results of the best configurations in Tables 4.16, 4.17, and 4.14. When increasing the value of the threshold, also the value of the baseline increases, meaning that the situation becomes more "constant" and the changes in the stations' state become more rare. The results of the classifiers do not differ from what we saw in the majority of the experiments: all the types of event generation are able to extract the baseline. However, rules with higher confidence and support are not extracted.

Experiments with different time zones

Since in this discussion there are many common facts in the experiments and configurations tested, there is not much interest in the results of the experiments for all the time zones with the "AlmostEmpty" target. For this reason, only one representative "time zone" was tested. The results are shown in Table 4.18, and are not different from the cases we already discussed, as they show the same analogy on the results, with a slightly different value for the baseline if we compare this experiment to its correspondent (Table 4.12).

4.3 Other approaches

In this section we present the results of the experiments with the two strategies introduced in Section 3.5. First, we will present the results in the case where a

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Empty-AlmostEmpty_wrapped	10	0.5	5	0.925146	0.682968	0.451759	0.543808
Empty-AlmostEmpty_wrapped	10	0.7	1	0.956742	0.777317	0.787611	0.782430
Empty-AlmostEmpty_wrapped	10	0.7	5	0.924815	0.735949	0.372263	0.494430
Empty-AlmostEmpty_wrapped	10	0.8	1	0.905668	0.789858	0.061061	0.113358
Empty-AlmostEmpty_wrapped	10	0.8	5	0.903073	0.768131	0.026557	0.051340
Empty-AlmostEmpty_wrapped	Dist	0.5	1	0.943464	0.680149	0.807070	0.738194
Empty-AlmostEmpty_wrapped	Dist	0.5	5	0.936154	0.688725	0.645042	0.666168
Empty-AlmostEmpty_wrapped	Dist	0.7	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-AlmostEmpty_wrapped	Dist	0.7	5	0.939865	0.774299	0.551985	0.644510
Empty-AlmostEmpty_wrapped	Dist	0.8	1	0.941553	0.779828	0.568760	0.657777
Empty-Increase_wrapped	10	0.5	1	0.942707	0.675629	0.807600	0.735743
Empty-Increase_wrapped	10	0.5	5	0.927811	0.641936	0.608384	0.624710
Empty-Increase_wrapped	10	0.7	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Increase_wrapped	10	0.7	5	0.933374	0.734750	0.509182	0.601514
Empty-Increase_wrapped	10	0.8	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Increase_wrapped	10	0.8	5	0.915294	0.808641	0.186396	0.302959
Empty-Increase_wrapped	Dist	0.5	1	0.942714	0.675585	0.807882	0.735835
Empty-Increase_wrapped	Dist	0.5	5	0.928428	0.643708	0.616542	0.629832
Empty-Increase_wrapped	Dist	0.7	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Increase_wrapped	Dist	0.7	5	0.934546	0.736069	0.525745	0.613378
Empty-Increase_wrapped	Dist	0.8	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Increase_wrapped	Dist	0.8	5	0.915433	0.808679	0.188233	0.305383
Empty-Decrease_wrapped	10	0.5	1	0.941790	0.669752	**0.809966**	0.733216
Empty-Decrease_wrapped	10	0.5	5	0.932021	0.650644	0.673047	0.661656
Empty-Decrease_wrapped	10	0.7	1	0.956742	0.777317	0.787611	0.782430
Empty-Decrease_wrapped	10	0.7	5	0.939579	0.751741	0.579602	0.654543
Empty-Decrease_wrapped	10	0.8	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Decrease_wrapped	10	0.8	5	0.918771	0.809941	0.231918	0.360586
Empty-Decrease_wrapped	10	0.9	1	0.902183	**0.897059**	0.010771	0.021287
Empty-Decrease_wrapped	0	0.5	1	0.955190	0.762365	0.793650	0.777693
Empty-Decrease_wrapped	0	0.5	5	0.917627	0.745866	0.251660	0.376340
Empty-Decrease_wrapped	0	0.7	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Decrease_wrapped	0	0.7	5	0.904241	0.742938	0.046440	0.087416
Empty-Decrease_wrapped	0	0.8	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Decrease_wrapped	Dist	0.5	1	0.941884	0.670370	0.809648	0.733456
Empty-Decrease_wrapped	Dist	0.5	5	0.938592	0.665625	0.759959	0.709671
Empty-Decrease_wrapped	Dist	0.7	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Decrease_wrapped	Dist	0.7	5	0.951629	0.776964	0.715638	0.745041
Empty-Decrease_wrapped	Dist	0.8	1	**0.957715**	0.785634	0.786411	**0.786022**
Empty-Decrease_wrapped	Dist	0.8	5	0.927326	0.793133	0.357324	0.492684
Dummy	0	0.0	1	**0.957715**	0.785634	0.786411	**0.786022**

Table 4.14: Results of the experiments with Time interval = 30 and target state = "AlmostEmpty".

classifier is built for each single station, then the results in the more general type of pre-processing. Finally, we make a comparison of the best performing strategies in each presented case to make a final evaluation of the adopted strategies.

4.3.1 Station-specific Approach

Some tests were made with this approach. First, a classifier with default parameters was trained for each considered type (Decision Tree, Random Forest and XGBoost).

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Empty-AlmostEmpty_wrapped	10	0.7	5	0.944670	0.818424	0.467765	0.595293
Empty-AlmostEmpty_wrapped	10	0.8	1	0.975300	0.855373	0.861782	0.858566
Empty-AlmostEmpty_wrapped	10	0.8	5	0.941912	0.832732	0.415815	0.554665
Empty-AlmostEmpty_wrapped	10	0.9	1	0.917440	0.875923	0.059402	0.111259
Empty-AlmostEmpty_wrapped	10	0.9	5	0.914791	0.879911	0.023781	0.046310
Empty-AlmostEmpty_wrapped	Dist	0.8	1	0.973415	0.835739	0.864287	0.849773
Empty-AlmostEmpty_wrapped	Dist	0.8	5	0.958643	0.858473	0.628160	0.725476
Empty-Increase_wrapped	10	0.7	1	0.974491	0.845335	0.865048	0.855078
Empty-Increase_wrapped	10	0.8	1	0.975629	0.858586	0.861802	0.860191
Empty-Increase_wrapped	10	0.8	5	0.940774	0.813975	0.413772	0.548647
Empty-Increase_wrapped	10	0.9	1	0.913406	0.883333	0.005309	0.010555
Empty-Increase_wrapped	Dist	0.7	1	0.974515	0.845499	0.865148	0.855211
Empty-Increase_wrapped	Dist	0.8	1	0.975624	0.858506	0.861842	0.860171
Empty-Increase_wrapped	Dist	0.8	5	0.939671	0.813370	0.397804	0.534295
Empty-Decrease_wrapped	10	0.8	1	0.975148	0.853545	0.862283	0.857892
Empty-Decrease_wrapped	10	0.8	5	0.952427	0.834334	0.565433	0.674055
Empty-Decrease_wrapped	10	0.9	1	0.917670	0.873570	0.062708	0.117016
Empty-Decrease_wrapped	10	0.9	5	0.914849	0.880029	0.024542	0.047753
Empty-Decrease_wrapped	0	0.5	1	0.975073	0.852022	0.863425	0.857686
Empty-Decrease_wrapped	0	0.5	5	0.922786	0.831287	0.141063	0.241196
Empty-Decrease_wrapped	0	0.7	1	0.975073	0.852022	0.863425	0.857686
Empty-Decrease_wrapped	0	0.7	5	0.922786	0.831287	0.141063	0.241196
Empty-Decrease_wrapped	0	0.8	1	**0.975803**	0.860750	0.861181	**0.860966**
Empty-Decrease_wrapped	0	0.8	5	0.914057	0.851981	0.014645	0.028795
Empty-Decrease_wrapped	Dist	0.8	1	0.973290	0.834213	**0.864848**	0.849254
Empty-Decrease_wrapped	Dist	0.8	5	0.966367	0.854881	0.738811	0.792619
Empty-Decrease_wrapped	Dist	0.9	1	0.918308	**0.890028**	0.069560	0.129035
Dummy	0	0.0	1	**0.975803**	0.860750	0.861181	**0.860966**

Table 4.15: Results of the experiments with Time interval = 15 and target state = "AlmostEmpty".

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Empty-AlmostEmpty_wrapped	Dist	0.5	1	**0.976025**	0.605085	0.606614	**0.605849**
Empty-Increase_wrapped	10	0.5	1	**0.976025**	0.605085	0.606614	**0.605849**
Empty-Increase_wrapped	10	0.7	1	0.973741	0.688981	0.246986	0.363621
Empty-Increase_wrapped	10	0.7	5	0.972478	0.674786	0.181307	0.285818
Empty-Increase_wrapped	Dist	0.5	1	**0.976025**	0.605085	0.606614	**0.605849**
Empty-Increase_wrapped	Dist	0.7	1	0.973741	0.688981	0.246986	0.363621
Empty-Increase_wrapped	Dist	0.7	5	0.971865	**0.691071**	0.133310	0.223506
Empty-Decrease_wrapped	10	0.5	1	0.975223	0.588313	**0.613848**	0.600809
Empty-Decrease_wrapped	10	0.7	1	0.973741	0.688981	0.246986	0.363621
Empty-Decrease_wrapped	10	0.7	5	0.972395	0.688152	0.166724	0.268417
Empty-Decrease_wrapped	0	0.5	1	**0.976025**	0.605085	0.606614	**0.605849**
Empty-Decrease_wrapped	0	0.7	1	0.973741	0.688981	0.246986	0.363621
Empty-Decrease_wrapped	Dist	0.7	1	0.973741	0.688981	0.246986	0.363621
Empty-Decrease_wrapped	Dist	0.7	5	0.973050	0.682528	0.210816	0.322134
Dummy	0	0.0	1	**0.976025**	0.605085	0.606614	**0.605849**

Table 4.16: Results of the experiments with Time interval = 30 and AC-TH = 1.

Later, since the overall results were low, a grid-search was done for each classifier. In Tables 4.19, 4.20, and 4.21 all the considered parameters are shown. Each grid search was done 3 times maximizing 3 different objective metrics: recall, precision,

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Empty-AlmostEmpty_wrapped	Dist	0.7	1	**0.966982**	0.712817	0.714069	**0.713443**
Empty-Increase_wrapped	10	0.7	1	**0.966982**	0.712817	0.714069	**0.713443**
Empty-Increase_wrapped	10	0.7	5	0.950952	0.763436	0.214312	0.334674
Empty-Increase_wrapped	10	0.8	1	0.942819	0.768473	0.009452	0.018675
Empty-Increase_wrapped	Dist	0.7	1	**0.966982**	0.712817	0.714069	**0.713443**
Empty-Increase_wrapped	Dist	0.7	5	0.950816	0.762801	0.211222	0.330834
Empty-Decrease_wrapped	10	0.7	1	**0.966982**	0.712817	0.714069	**0.713443**
Empty-Decrease_wrapped	10	0.7	5	0.953456	0.719284	0.313924	0.437086
Empty-Decrease_wrapped	10	0.8	1	0.945442	0.775785	0.073376	0.134071
Empty-Decrease_wrapped	10	0.8	5	0.942446	**1.000000**	0.000121	0.000242
Empty-Decrease_wrapped	0	0.7	1	**0.966982**	0.712817	0.714069	**0.713443**
Empty-Decrease_wrapped	Dist	0.5	1	0.956044	0.595777	**0.735155**	0.658168
Empty-Decrease_wrapped	Dist	0.7	1	**0.966982**	0.712817	0.714069	**0.713443**
Empty-Decrease_wrapped	Dist	0.8	1	0.945762	0.770278	0.082283	0.148683
Dummy	0	0.0	1	**0.966982**	0.712817	0.714069	**0.713443**

Table 4.17: Results of the experiments with Time interval = 30 and AC-TH = 2.

Extraction Type	#neighbors	Conf Thr	Match Thr	Accuracy	Precision	Recall	F1
Empty-AlmostEmpty_wrapped	10	0.7	1	0.903835	0.709677	0.006335	0.012557
Empty-AlmostEmpty_wrapped	Dist	0.5	1	**0.946859**	0.678727	0.853441	**0.756122**
Empty-Increase_wrapped	Dist	0.5	1	**0.946859**	0.678727	0.853441	**0.756122**
Empty-Decrease_wrapped	10	0.5	1	0.945553	0.671035	**0.855168**	0.751994
Empty-Decrease_wrapped	10	0.7	1	0.904169	**0.837838**	0.008926	0.017664
Empty-Decrease_wrapped	0	0.5	1	**0.946859**	0.678727	0.853441	**0.756122**
Empty-Decrease_wrapped	0	0.5	5	0.906087	0.755435	0.040023	0.076019
Empty-Decrease_wrapped	Dist	0.5	1	**0.946859**	0.678727	0.853441	**0.756122**
Dummy	0	0.0	1	**0.946859**	0.678727	0.853441	**0.756122**

Table 4.18: Results of the experiments in the time zone 16 - 20.

and f1. These experiments were done twice, considering time intervals of 30 minutes with 5 and 3 time windows. The best results of each classifier are shown in Tables 4.22 and 4.23. In both cases the highest results in terms of precision and f1 score are reached by the XGBoost classifier.

4.3.2 Station-non-specific Approach

With this type of approach, the same classifiers and grid-search parameters as in Section 4.3.1 were considered. In this case, time intervals of 30 minutes, and 3 consecutive time windows were tested for the whole day and the time zones considered in the experiments for the associative classifier. The best results of each classifier are shown in Table 4.24. In this case the best results of f1 are achieved by the XGBoost classifier, whereas Random Forest achieved higher precision. Since the experiments conducted on each separate time zone did not demonstrate different results than the one just presented, we omit such results in this chapter.

Parameter	Values
Max Depth	2, 3, 4, 5
Min samples split	2, 3, 4, 5, 6
Criterion	gini, entropy
Class weight	none, balanced

Table 4.19: Grid-search parameters of the Decision Tree classifier.

Parameter	Values
Max Depth	2, 3, 4, 5
N. of estimators	10, 100, 1000
Min samples split	0, 1, 2, 3, 4
Criterion	gini, entropy
Class weight	none, balanced

Table 4.20: Grid-search parameters of the Random Forest classifier.

4.4 Overall results

Up to this point, the most relevant experiments and result have been presented. To make a final comparison, we can pick the tests with intervals of 30 minutes and 3 consecutive time windows, that were tested for all the considered models. The results are summarized in Table 4.25. As we can see, the associative classifier is not the best performing model if we consider the f1 score, however the difference from the best one (XGBoost) is below 2%. If we consider the precision instead, the configurations of the associative classifier considering the "negative" states significantly outperform the other classifiers ($\approx 7\%$ more). We can conclude that the associative classification is a valid option if we are interested in models that are extremely reliable when making a prediction.

Parameter	**Values**
Max depth	2, 3, 4, 5
N. of estimators	10, 100, 1000

Table 4.21: Grid-search parameters of the XGBoost classifier.

Classifier	**Avg Accuracy**	**Avg Recall**	**Avg Precision**	**Avg F1_score**
DecisionTree	0.981	0.768	0.721	0.744
RandomForest	0.961	**0.803**	0.473	0.595
XGBoost	**0.983**	0.750	**0.764**	**0.757**

Table 4.22: Best classifiers results considering 5 consecutive time windows.

Classifier	**Avg Accuracy**	**Avg Recall**	**Avg Precision**	**Avg F1_score**
DecisionTree	0.981	0.755	0.721	0.737
RandomForest	0.964	0.**832**	0.494	0.620
XGBoost	**0.983**	0.758	**0.763**	**0.760**

Table 4.23: Best classifiers results considering 3 consecutive time windows.

Classifier	**Accuracy**	**Recall**	**Precision**	**F1_score**
DecisionTree	**0.978**	0.769	0.868	**0.816**
RandomForest	0.976	0.731	**0.874**	0.796
XGBoost	**0.978**	**0.771**	0.867	**0.816**

Table 4.24: Best classifiers results considering 3 consecutive time windows in the whole day

Classifier	**Accuracy**	**Precision**	**Recall**	**F1**
Full-AlmostFull_wrapped	0.974	0.793	**0.794**	0.794
Full-Decrease_wrapped	0.974	0.793	**0.794**	0.794
Full-Increase_wrapped	0.974	0.793	**0.794**	0.794
Full-Increase_wrapped_neg	**0.978**	0.943	0.696	0.801
Full-Decrease_wrapped_neg	**0.978**	**0.944**	0.695	0.800
Dummy	0.974	0.793	**0.794**	0.794
DecisionTree	**0.978**	0.868	0.769	**0.816**
RandomForest	0.976	0.874	0.731	0.796
XGBoost	**0.978**	0.867	0.771	**0.816**

Table 4.25: Comparison of all the best classifiers tested.

Chapter 5

Conclusions

5.1 Results

In these chapters we discussed the formulation of the problem, the analysis of the data, the proposed solution and its results. We analyzed the available dataset, and built the association classifier based on spatio-temporal contextual information.
The results of the classification show that our associative classifier is not able to improve the results of the other models tested in terms of f1 score. However, all the best classifiers tested reach values in range 80% - 81%, and none of them significantly outperforms the baseline. If we consider the precision, the associative classifier is able to achieve the best results (94%), 7% higher than the second best model.
In our context, a classifier with such a high precision score can provide valuable insights to bike sharing managers, because highly reliable predictions can enable optimized and targeted balancing operations. For this reason, we can be satisfied with the results achieved by the associative classifier.

5.2 Future Works

In this work, plenty of experiments were conducted, and many kinds of event generation were tested. However, the generations considering also the "not increasing" and "not decreasing" events, that obtained the most promising results, were tested only in few of the experiments. Specifically, further tests on the most relevant time zones need to be run considering the two aforementioned events.
Additionally, we plan to extend the proposed methodology in the multi-class classification context, considering Critical and Almost Critical as two separate cases, with the goal of obtaining more specific results.

Bibliography

[1] Elena Baralis. *The Big Data challenge*. URL: `https://dbdmg.polito.it/dbdmg_web/wp-content/uploads/2021/09/1-DSIntro.pdf` (cit. on p. 3).

[2] Matei Zaharia et al. «Apache spark: A unified engine for big data processing». In: *Communications of the ACM* 59 (Nov. 2016), pp. 56–65. DOI: `10.1145/2934664` (cit. on p. 4).

[3] Elena Baralis. *Association Rules Fundamentals*. URL: `https://dbdmg.polito.it/dbdmg_web/wp-content/uploads/2021/10/7-DMassrules.pdf` (cit. on pp. 4, 5).

[4] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. «Mining frequent patterns without candidate generation: A frequent-pattern tree approach». In: *Data mining and knowledge discovery* 8.1 (2004), pp. 53–87 (cit. on p. 5).

[5] Matteo Golfarelli. *Pattern sequenziali*. URL: `http://bias.csr.unibo.it/golfarelli/DataMining/MaterialeDidattico/2017/12-Pattern%20sequenziali.pdf` (cit. on p. 5).

[6] Jian Pei, Jiawei Han, B. Mortazavi-Asl, H. Pinto, Qiming Chen, U. Dayal, and Mei-Chun Hsu. «PrefixSpan: mining sequential patterns efficiently by prefix-projected pattern growth». In: (2001), pp. 215–224. DOI: `10.1109/ICDE.2001.914830` (cit. on p. 5).

[7] Francesco Vaccarino. *Lecture on Classification* (cit. on p. 6).

[8] Elena Baralis. *Classification Fundamentals*. URL: `https://dbdmg.polito.it/dbdmg_web/wp-content/uploads/2021/10/8-DMClassification.pdf` (cit. on p. 6).

[9] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. URL: `http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning` (cit. on pp. 6, 7).

[10] Francesco Vaccarino. *Lecture on Tree-Based Methods* (cit. on p. 7).

[11] Tianqi Chen and Carlos Guestrin. «XGBoost: A Scalable Tree Boosting System». In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: Association for Computing Machinery, 2016, pp. 785–794. ISBN: 9781450342322. DOI: `10.1145/2939672.2939785`. URL: `https://doi.org/10.1145/2939672.2939785` (cit. on p. 8).

[12] Ben Hamner. *SF Area bike share*. URL: `https://www.kaggle.com/benhamner/sf-bay-area-bike-share` (cit. on p. 11).